COMMODORE & AMIGA nr 9



NR INDEKSU 355216
ISSN 0867-8022
Cena 10 000 zł
wrzesień 1992 r.

MAGAZYN UŻYTKOWNIKÓW KOMPUTERÓW «COMMODORE»



- ZOSTAŃ „WŁAMYWACZEM"
- TRACK BALL TKB-MT-AC
- NEWIO
- MUZYKA I C-128

... i wiele innych

MILLENNIUM PRESENTS
A FUTURISTIC GAME OF STRATEGIC CONFLICT

MILLENNIUM

Na planecie Orion o dominację walczy pięć potęg militarno-przemysłowych. Jako Naczelny Dowódca jednej z nich masz za zadanie ułożyć plan i poprowadzić kampanię strategiczną w wyniku której pokonasz rywali i zdobędziesz władzę nad planetą. Buduj fortyfikacje i zakłady zbrojeniowe, powiększaj swe terytoria, prowadź wojnę z rywalami. Twoje rozkazy to KONSTRUUJ, DOWODŹ, ZDOBYWAJ!



COMPUTER GROUP

Tą a także ponad 50 innych gier firmy Electronic Arts, Domark, Virgin, Sierra on-Line kupisz u nas i w ponad 100 innych sklepach na terenie całego kraju. Zadzwoń do nas, a dowiesz się, gdzie najbliżej Twego miejsca zamieszkania znajdziesz oferowane przez nas gry.

UL. OKRĘŻNA 3 02–916 WARSZAWA TEL.: (02) 642–27–66 TEL.: (02) 642–27–68 FAX: (02) 642–27–69 TLX: 81 6351 IPS–PL

W chwili, gdy numer ten ukaże się w kioskach będziecie prawdopodobnie dzielili się pierwszymi wrażeniami ze szkoły. Sądząc po zeszłorocznych zabiegach Ministerstwa Edukacji Narodowej i redakcjach nauczycieli głównym problemem z jakim boryka się szkolnictwo państwowe jest brak pieniędzy na podwyżki dla pedagogów. Pesymiści twierdzą, że niebawem szkolnictwo państwowe będzie miało jeszcze jeden problem — brak wykwalifikowanej kadry, która pójdzie do jakiejkolwiek innej pracy zapewniającej przeżycie.
Tego rodzaju problemów nie mają szkoły prywatne — czesne i opłaty miesięczne są tam wyśrubowane do granic możliwości. Szkolnictwo prywatne boryka się jednak z problemami natury lokalowej co wydaje się być jeszcze trudniejsze do załatwienia aniżeli podwyżki. Przeprowadzona ostatnio reforma (?) szkolnictwa polegała głównie na wprowadzeniu dwóch nowych przedmiotów — religii i etyki. Ze swojej strony nie mam nic przeciwko nim, zastanawia mnie jednak koszmarna opieszałość we wprowadzeniu podstaw posługiwania się komputerem. Efekty tej niefrasobliwości zbierają już w tej chwili absolwenci szkół średnich przed którymi zamknęły się możliwości studiowania; pierwsze dwa pytania zadawane obecnie przez pracodawców dotyczą znajomości języka oraz umiejętności posługiwania się komputerem, telefaksem i kserografem. Jeśli odpowiedź kandydata do biurka brzmi negatywnie skutki łatwo przewidzieć.
Niektóre szkoły nie czekając na wytyczne same zakupiły sprzęt, same zorganizowały odpowiednie klasy, same zatrudniły specjalistów. W efekcie liczba osób na jedno miejsce jest w tych placówkach bardzo wysoka; istnieje bowiem uzasadnione przekonanie, że absolwent takiej szkoły, jeśli nawet nie dostanie się na studia, zawsze znajdzie jakąś pracę „na przeczekanie". Takiej gwarancji nie daje natomiast większość szkół, których nie było stać na zakup sprzętu czy zatrudnienie nauczycieli.
W niektórych krajach obsługi komputera uczy się niemal od pierwszej klasy szkoły podstawowej, dzięki czemu kolejne generacje pracowników nie dostają spazmów i rozstroju nerwowego na widok klawiatury, nie muszą również zapisywać się na kursy komputerowe. U nas widać, że prawda ta jeszcze nie dotarła do właściwego urzędnika, a może ktoś celowo posadził tam głuchego faceta.

KLAUDIUSZ DYBOWSKI

M·E·N·U

magazyn użytkowników komputerów «COMMODORE»



Redaktor naczelny: KLAUDIUSZ DYBOWSKI
Sekretarz redakcji: CHRISTIAN GRZENKOWICZ
Opracowanie graficzne: JOLANTA PRZEŹDZIECKA
Redaktor techniczny: JOLANTA SZUMOWSKA
Zdjęcia: JERZY STOKOWSKI

Stali współpracownicy: ANDRZEJ BOBEK (szef Działu Amigi)
BARTŁOMIEJ DRAMCZYK
JERZY DUDEK
MARIUSZ FERDYN
BARTŁOMIEJ KACHNIARZ
WOJCIECH KAZIMIERCZAK
PIOTR LISZEWSKI
RAFAŁ PIASEK
BARTOSZ SMAGA
RAFAŁ WIOSNA

Redakcja: ul. Wasilkowskiego 7, 02-776 WARSZAWA tel./BBS: 643-1840
Kontakt z Czytelnikami: pon-pt w godzinach 10.00-17.00
Wydawca: Spółdzielnia „Bajtek" ul. Wspólna 61 00-687 Warszawa tel./fax: 21-12-05
Skład i druk: Przedsiębiorstwo Wydawniczo-Poligraficzne „GRYF", Sp. Akc. Ciechanów
Korekta: KRYSTYNA WYDURSKA, MARIA GOŹDZIEWSKA
Nr zlecenia: 43832
Nakład: 72 tys. egzemplarzy



WYMIANA PROCESORA 68000 na 68010 W AMIDZE

Kiedyś spróbowałem zainstalować mikroprocesor 68010 w mojej Amidze. Miałem z tym trochę kłopotów, ale uzyskany efekt wynagrodził mi trudy tego zadania. Teraz mogę gorąco polecić wymianę mikroprocesora każdemu. Jeżeli chcesz dokonać przeróbki swojej Amigi, to musisz przeczytać i zrozumieć cały artykuł!

MC 68010 (wykonany w technologii DIL — Dual In Line) ma identyczny układ wyprowadzeń jak 68000. Przeróbka sprzętowa polega więc na zamianie obu procesorów.

UWAGA!:
WNĘTRZE AMIGI JEST BARDZO WRAŻLIWE NA ŁADUNKI ELEKTROSTATYCZNE. Z KOMPUTEREM NALEŻY WIĘC OBCHODZIĆ SIĘ BARDZO OSTROŻNIE. NALEŻY PAMIĘTAĆ, ŻE OTWARCIE KOMPUTERA POWODUJE UTRATĘ PRAW GWARANCYJNYCH. JEŻELI NIE CZUJESZ SIĘ NA SIŁACH — POPROŚ O POMOC FACHOWCA. I PAMIĘTAJ, NIE ŚPIESZ SIĘ.

Przeróbkę należy rozpocząć od odłączenia Amigi od wszystkich urządzeń peryferyjnych. Następnie odkręć sześć śrub znajdujących się w dolnej części wokół całej obudowy. Zdejmij górną część obudowy i odłóż na bok. Delikatnie odłącz pęk przewodów łączących klawiaturę z płytą główną od gniazda znajdującego się na płycie. Ostatnim etapem jest usunięcie metalowej osłony antyradiacyjnej, przymocowanej w kilku miejscach śrubkami, a w kilku przy pomocy zaginanych języczków z blachy.

Po tej operacji masz przed oczami płytę główną Amigi. Zlokalizuj teraz mikroprocesor 68000 — jest to duża kość (64 nóżki) znajdująca się po lewej stronie, w pobliżu gniazda systemowego. Zwróć uwagę z której strony znajduje się półokrągłe nacięcie w obudowie — określa ono właściwe położenie kości w podstawce.

Spróbuj teraz wyjąć układ z podstawki (bardzo ostrożnie podważając go) i w jego miejsce włóż nowy procesor 68010 (zwracając uwagę na nacięcia w podstawce i w układzie). Usunięty z płyty 68000 najlepiej zawiń w folię aluminiową (w celu ochrony przed napięciami statycznymi) i zachowaj, gdyż może przydać się później. Teraz podłącz z powrotem wszystkie przewody i klawiaturę, lecz nie skręcaj jeszcze Amigi.

Włącz zasilanie i włóż do stacji dysków dysk z Workbenchem. Wczytaj kalkulator i wykonaj operację "9x3=". Jeśli w tym momencie Amiga zaprotestowała znajomą ramką z napisem „Software Failure" to znaczy, że nowy procesor działa. Dzieje się tak dlatego, ponieważ kalkulator korzysta z instrukcji MOVE SR,‹ea›, zabronionej w trybie użytkownika (User Mode) w 68010. Możesz teraz spokojnie zacząć czytać drugą część poświęconą stronie programowej przeróbki. Oczywiście po skręceniu Amigi.

Jeżeli Twoja Amiga zachowuje się dziwnie, np. miga dioda Power to znaczy, że zainstalowałeś niesprawny procesor lub uszkodziłeś w czasie przeróbki inny układ. Spróbuj wtedy włożyć z powrotem stary procesor 68000 i sprawdź, czy system działa. Jeżeli nie, to musisz skontaktować się z serwisem zajmującym się Amigą.

PROCESOR 68010 OD STRONY PROGRAMOWEJ

Jak zapewne zauważyłeś, po zmianie procesora na 68010 możesz korzystać z większości oprogramowania bez problemu. Niestety niektóre programy (korzystające z rozkazu MOVE SR,‹ea›) będą powodowały zawieszanie się systemu i pojawienie się komunikatu Guru.

Na Twoje szczęście istnieje rozwiązanie i tego problemu: DeciGel. Programik ten, umieszczony w pamięci jako program rezydentny, przechwytuje zakazaną instrukcję i zastępuje ją w pamięci instrukcją MOVE CCR,‹ea›, po czym wykonywanie programu jest kontynuowane. Podczas testowania programów, które wcześniej sprawiały kłopoty okazało się, że DeciGel zapewnia 100% kompatybilność ze starym MC 68000! Teraz wszystkie programy działają bez zarzutu.

WYMIENIAĆ CZY NIE WYMIENIAĆ?

Amiga 500 z mikroprocesorem 68010 i programem DeciGel działa z szybkością od 4% do 50% większą, w zależności od rodzaju operacji. Największe przyspieszenia uzyskuje się w programach, w których procesor wykonuje dużo obliczeń np. w cruncherach, programach „pakujących", programach do DTP czy grafiki. Wybór należy do Ciebie...

JAK ZAINSTALOWAĆ DECIGEL?

Najlepiej umieścić go w katalogu C i wpisać jego nazwę w Startup-Sequence na samym początku. Operację tę należy powtórzyć na wszystkich dyskietkach zawierających programy sprawiające kłopoty z nowym procesorem. Program DeciGel znajduje się natomiast na wrześniowej dyskietce C&A oznaczonej "A09/92".

opracował

JERZY „BLUE THUNDER" DUDEK



NEWIO

ARKUSZ DO PROJEKTOWANIA PŁYTEK DRUKOWANYCH

Jednym z bardziej przyjaznych programów do projektowania płytek drukowanych jest dla Amigi program NEWIO. Właściwie nie jest to program projektujący tylko arkusz, na którym użytkownik sam rysuje połączenia między elementami, sam rozmieszcza elementy na płytce i decyduje o jej wielkości. Mimo braku automatyki, tak jak ma to miejsce w programach Pro-Board czy OrCAD (dla komputerów PC) jest to bardzo przydatne narzędzie. Większość płytek projektowanych przeze mnie powstaje właśnie na NEWIO.

Program zawiera mnóstwo gotowych szablonów takich jak podstawki pod układy scalone, począwszy od 6. nóżkowej (np. dla transoptorów CNY17), a na 64. nóżkowej (np. dla procesora Motorola 68000) skończywszy. Dla tranzystorów również zawarto całą gamę szablonów dla różnych obudów np. TO-3, SOT-92 i wiele innych. Elektroników zajmujących się techniką komputerową ucieszy fakt zawarcia w programie szeregu złącz: od Cannon 9-stykowego do krawędziowych 96-stykowych. Nie pominięto także elementów biernych. Mamy szablony dla rezystorów, kondensatorów, trymerów i potencjometrów, oczywiście o różnych rozstawach nóżek. Jeśli chodzi o pojedyncze punkty lutownicze, to możemy ustawiać ich średnicę od 1.0 mm do kilku milimetrów ze skokiem 0.6 mm. Grubość ścieżek jest także regulowana. Możemy programowi zabronić ustawiania ścieżek nieprostopadłych bądź nierównoległych do siebie, możemy zezwolić na kąt 45 stopni, bądź wybrać dowolny kąt.

Edycja wykonanej płytki jest również bardzo wygodna. Możemy wyciąć dowolny fragment, skopiować go do bufora, przesunąć czy powielić. Na ekranie widoczne są trzy rysunki nałożone na siebie: czerwony — strona elementów, zielony — strona lutowania i niebieski — strona opisu. Fragmenty pokrywające się ze sobą widoczne są w kolorze żółtym (np. punkt istniejący w tym samym miejscu na stronie lutowania i stronie elementów będzie widoczny w tym właśnie kolorze). Jeśli chodzi o wykonanie strony opisowej to mamy możliwość wyboru wielkości tekstu i kąta jego ustawienia.

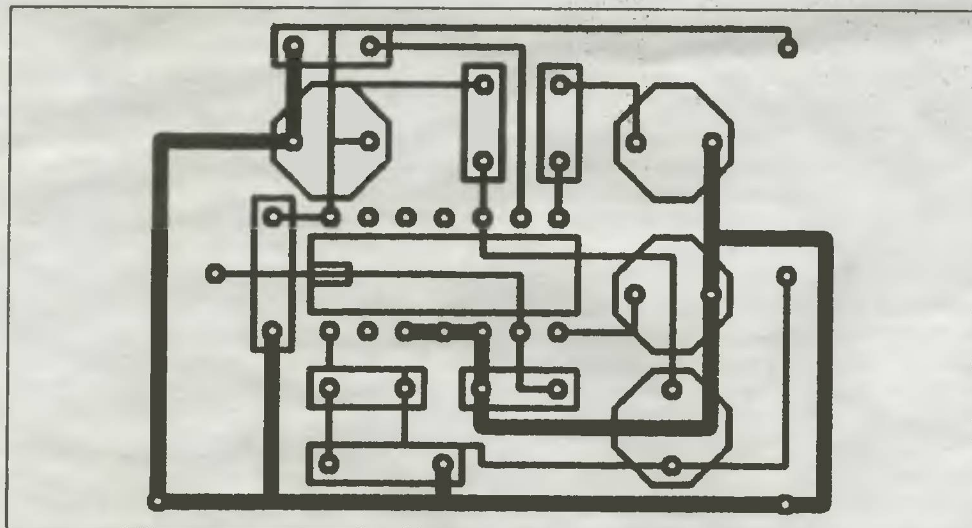
Gdy przeszkadza nam widok jednocześnie trzech rysunków na ekranie, mamy możliwość przyciemnienia bądź całkowitego wyłączenia niepotrzebnych stron (Element-Path--Marker Side). Na stronie opisowej (marker side) widoczna jest siatka (grid), którą również można wyłączyc. Ułatwia ona zachowanie prostopadłości i równoległości ścieżek.

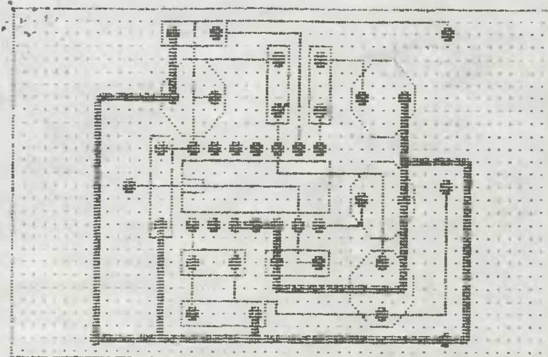
Bardzo dużą zaletą programu jest opcja wydruku. Jest to w zasadzie oddzielny program na tej samej dyskietce pod nazwą Newio-Print. Współpracuje on z plotterem i drukarkami igłowymi, ale tylko kompatybilnymi z EpsonX bądź NEC P6. Nie udało mi się namówić programu (jak do tej pory) do współpracy z drukarką laserową.

Uzytkownik ma możliwosc wydruku dowolnej strony płytki w skali 1:1, 2:1 bądź 4:1. Możemy także wydrukować dwie lub trzy strony nałożone na siebie. Wydruk jest bardzo wysokiej jakości i nadaje się bezpośrednio do fotograficznej metody wytrawiania płytek. Dodatkową zaletą jest możliwość wydrukowania tzw. „solder maski" i mapy otworów dla wiertarki numerycznej (!).

Podsumowując, należy program ocenić jako przydatny w pracy elektronika. Nieco dziwny jest sposób wyboru funkcji (brak rozwijanych menu), ale można przymknąć na to oko, biorąc pod uwagę niezaprzeczalne zalety programu. Testowana przeze mnie wersja nosi numer 1.0 i jest to wersja angielska (istnieje także niemiecka).

JERZY „BLUE THUNDER" DUDEK





Przykładowe wydruki



AMOS

Ostatnio bardzo głośno jest o tym nowym języku. Nawet na giełdach, pytają o AMOS. Tajemniczy AMOS. Czym właściwie jest?

Reakcje wśród użytkowników są podzielone. Ludzie, będący poza tematem, obsługujący stale coraz nowsze wersje X-Copy, (nawiasem mówiąc są to swego rodzaju konie trojańskie), ci to właśnie śmiałkowie wzruszają nań ramionami. Tylko jeden z moich znajomych, zaliczający się do tej właśnie komputerowej „kasty" zapytał, czy AMOSem przypadkiem nie można szybciej czegoś skopiować. Gdy otrzymał okraszoną epitetami odpowiedź postąpił jak inni.

Nieco inaczej odnoszą się do AMOSa programujący w języku C. Jeden z nich, oglądając moje wypociny powiedział, że „zrobi to szybciej". Najlepszymi cwaniakami są jednak koderzy. Twarz tych ludzi, zawsze nienagannie spokojna, kryje teraz pod szyderczym uśmiechem nutkę strachu. Myśli krążą wokół jednego: „Ludzie, a co będzie gdy stanę się już nikomu niepotrzebny?" i po nocach dręczą ich koszmary w stylu: „Świat opanowany jest przez AMOSowców".

Żarty żartami, ale AMOS to naprawdę coś nowego i dość trudno sprecyzować co to właściwie jest. Krzywdzącym, ale bliskim prawdy jest określenie — język wysokiego poziomu, a już przekleństwem — BASIC. Jest to niestety prawda, która cieszyć powinna chętnych do nauki programowania.

Twórcą języka AMOS jest Francois Lionet; pierwowzorem języka był STOS dla komputerów Atari ST. Najnowszą wersją AMOSA jest AMOS V1.2 produkowana przez firmę MANDARIN SOFTWARE. Sam język ma już pakiety dodatkowe — jest to kompilator (AMOS COMPILER) oraz program o nazwie AMOS 3D. Dokładniej zajmiemy się nimi później, powiem tylko, że AMOS 3D to edytor grafiki wektorowej języka. Tu na marginesie pewne wyjaśnienie. W dokumentacji od kompilatora napisano, że wymaga on wersji interpretera 1.3. Skąd go w takim razie wziąć, skoro takiej wersji w ogóle nie ma? Rozwiązanie problemu jest proste. Zarówno w programie instalującym kompilator, jak i na dyskietce z AMOS 3D jest plik „Update" czyli program uaktualniający wersję 1.2 interpretera do wersji 1.3.



Z najświeższych wiadomości należałoby dodać, że powstał już nowy interpreter — Easy AMOS. Jestem od wczoraj posiadaczem także i tej wersji, ale niestety nie mam od niej żadnej dokumentacji. Adresatem tego AMOSa powinni być jednak przede wszystkim początkujący. Easy AMOS ma wbudowany tzw. debugger, który umożliwia wykonanie programu krok po kroku, kontrolując stan ekranu, zmiennych i parametrów wykonywanych instrukcji. Pozwala to szybko zlokalizować ewentualne błędy. Zmieniono również grafikę interpretera i poprawiono requestery, o których za chwilę. Rozbudowano także system ekranów pomocniczych i wbudowano niektóre edytory.

Cały produkt rozprowadzany jest na dwóch dyskietkach instalacyjnych, a sama instalacja przypomina niemal oglądanie dema. Maskotką tego języka jest Baby Amos, który towarzyszy nam podczas całej procedury instalacyjnej. Nie będę wam już więcej psuł dobrej zabawy. Proponuję zdobyć ten program, a ja będę się starał, aby moje artykuły o AMOSie były uniwersalne. W najbliższym czasie opiszę zawartość dysków z Easy AMOSem i będę sprawdzał także jego kompatybilność ze standardem (niestety już z pierwszych oględzin wynika, że nie są one w pełni zgodne).

AMOS pozwala na kontrolowanie pracy praktycznie całego komputera. Mamy dostęp do rejestrów procesora, bibliotek (Dos, Exec, Intuition, Gfx itd.), możemy modyfikować Copper-List, a także dokonywać operacji dyskowych (odczytywanie katalogu dysku, ilości wolnego na nim miejsca, utworzenie katalogu, skasowanie istniejącego zbioru itd.). Tworzenie jakiejkolwiek grafiki, animacji może być zsynchronizowane z wyświetlaniem obrazu (Vertical Blank Interrupt), co pozwala na ich upłynnienie i uniknięcie związanych z tym niepożądanych efektów wizualnych. AMOS ma wbudowany specjalny język animacji — AMAL (AMOS Animation Language). Program napisany w tym języku umożliwia poruszanie jednego obiektu poczynając od sprite'a, a skończywszy na całym ekranie (screen). Programów tych może być wykonywanych jednocześnie aż 16 (a nawet i 64) i nie wpływają one na pracę samego AMOSa. Podobnie niezależnie od programu wykonują się operacje generowania dźwięku, oraz niektóre operacje graficzne, np. Fade — płynna zmiana palety kolorów, czy Appear — płynne przejście jednego obrazu w drugi. Daje to olbrzymie możliwości tworzenia rozbudowanych kombinacji graficzno-muzyczno-animacyjnych.

AMOS wyposażono w szybki, wygodny edytor, który umożliwia kopiowanie, wycinanie i wstawianie bloków tekstu. Ma on także okno trybu interakcyjnego, w którym mamy dostęp do wszystkich poleceń AMOSa z wyjątkiem pętli i procedur. Możliwa jest naprzemienna edycja wielu znajdujących się w pamięci programów, uruchamianie, testowanie i przenoszenie ich części między sobą.

Dla mnie prywatnie AMOS jest czymś, czego jeszcze nie było. Ma on jednak kilka rzeczy, które mnie denerwują. Własny zestaw „screenów", okien, requesterów nie pozwala na dopasowanie się do amigowskiego standardu. Jest to bardzo oryginalne, ale mnie osobiście nie zawsze odpowiada. Zupełnie tragiczny jest natomiast requester, zapamiętujący ścieżkę dopiero po wybraniu odpowiedniej opcji i za każdym razem wczytujący od nowa katalog. Posiadam dwie wersje AMOSa i nie są to oczywiście wersje firmowe. Nie wiem komu zawdzięczać mam brak gadgetów do wyboru urządzeń (wersja 1.2/1.3), które występują w wersji 1.1. Czy są za to odpowiedzialni polscy „hackerzy"? Requester ten często się blokuje wyświetlając w oknie urządzenia logiczne, których nie można w żaden sposób uaktywnić myszą.

Nie będą jednak tym problemem przejmować się ci, którzy mają Easy AMOS. Requestery zostały poprawione, a ich używanie jest bardzo podobne do tych „normalnych", amigowskich (Req.library, Reqtools.library). Drugą dziwną innowacją jest używanie przez AMOS niestandardowych opisów ścieżek i jokerów. Wszystkie te wady, to kwestia przyzwyczajenia i myślę, że nie tylko ja tak uważam.

Interpreter AMOSa zawiera bardzo wiele bardzo zintegrowanych instrukcji. Wykrycie przyciśnięcia klawisza obsługuje tylko „Wait Key", kliknięcie myszą — „Mouse Click". „Wait Vbl" poprzedzające instrukcję np. przesunięciu obrazu, powoduje czekanie na przerwanie kreślenia obrazu na monitorze w prawym dolnym punkcie ekranu. Teraz działo eketronowe kineskopu przestaje wysyłać elektrony i wraca w lewy górny róg, a w tym „martwym" dla grafiki czasie program wykonuje naszą instrukcję przesunięcia. Nowy obraz generowany jest już po wykonaniu tej instrukcji i oko ludzkie nie dostrzega na ekranie żadnego, załamania, mignięcia i innych nieprzyjem-



nych efektów. Jest to bardzo pomocne i daje efekt nawet przy kreśleniu kilku kresek. Wyobraźmy sobie jak wyglądałby listing programu wykonującego taką operację w jakimkolwiek innym języku! Dodać także należy, że wszystkie złożone instrukcje są wykonywane bardzo szybko, a program interpretowany w AMOSie ma prędkość porównywalną z kompilowanymi programami z innych języków.

AMOS ma też definiowalne banki pamięci. Możemy do nich załadować właściwie wszystko: sprite'y, boby, obrazy, muzykę, sample, kod maszynowy itd. Z moich własnych doświadczeń wiem, że język ten bardzo szanuje pamięć, co przy 1 MB RAM pamięci pozwala szaleć do woli (np. otworzyć wszystkie screeny, przeprowadzać operacje animacyjne i odtwarzać moduł muzyczny). Jeśli chodzi o odtwarzanie muzyki, to AMOS ma własny format zapisu. Istnieją konwertery różnych standardów muzycznych (Sonix, NoiseTracker, GMC) na format AMOSa. Mamy wtedy możliwość zmiany tempa, głośności, włączania i wyłączania filtru górnoprzepustowego, a także odczytywania głośności poszczególnych kanałów, co daje możliwość zbudowania w prosty sposób efektownego equalizera. Nawet jednak brak tych konwerterów (ja ich nie posiadam) nie wyklucza możliwości odtwarzania muzyki przez AMOS. Posługuję się tu procedurą odtwarzającą napisaną w asemblerze, którą po skompilowaniu „wciągnąłem" do banku pamięci AMOSa. Procedurę taką można znaleźć często na dyskietce z programem muzycznym, za POMOCĄ KTÓREGO stworzyliśmy moduł. Dodam tylko, że programy do konwersji opisanych formatów znajdują się w prawie każdej zachodniej bibliotece Public Domain i nie powinno być z ich zdobyciem żadnych problemów. Wyjątkiem jest Easy AMOS, który ma zaimplementowany zestaw instrukcji ładujących i odtwarzających moduły w formacie Noise Trackera.

Definiowanie screenów, okien, menu, gadgetów, suwaków, zmiana wyglądu wskaźnika myszy i inne operacje wymagające dobrej znajomości innego języka, w AMOSie załatwiamy dosłownie pojedynczymi instrukcjami. Jeśli chodzi o wskaźnik myszy to AMOS ma już gotowe cztery wzory jego wyglądu, 16 wzorów wyglądu okna, 35 wzorów wypełniania zamkniętej powierzchni. Można oczywiście wykorzystać własne. Najciekawsza jest chyba możliwość tworzenia ruchomego menu (główna belka screenu nie musi się wcale znajdować na jego szczycie!).

Aby zacząć programować w AMOSie radziłbym przede wszystkim (tu się zwracam do zupełnie początkujących) spróbować odrobinę pomajstrować w języku BASIC. AMOS wykona prawie każdy program z „małego" komputera. Oczywiście nie może on zawierać instrukcji „Poke", „Peek" itp. A co później? Można także zajrzeć do polskiej literatury. Ja korzystam z książki „Amiga Amos", opracowanej na podstawie wersji angielskiej. Książka ta jest dobra, ale zawiera sporo błędów, zwłaszcza w formatach instrukcji. Są to drobne, ale wykluczające ich poprawne użycie „chochliki". Mnie zabrało sporo czasu ich poprawienie, ale mój młodszy sąsiad zupełnie się załamał. Co najmniej dwa razy dziennie słyszę dzwonek do drzwi, a za drzwiami stoi szukający pomocy człowiek.

Język ten jest prosty dla tych, którzy wiedzą co znaczy Print, For, Next itd.: jest cała rzesza tych, którzy tego nie wiedzą. Dla zachęty mogę powiedzieć, że na scenie jest parę znanych „demotwórczych" grup, które „przerzuciły" się na AMOS. A ja od dwóch dni usuwam błąd, z zajmującego 60 linii programu w asemblerze, którego zadaniem jest sprawdzenie obecności dysku w stacji, otworzenie okna i wyświetlenie komunikatu. Zapewniam was, że taki sam program w AMOSie można prawdopodobnie napisać w jednej linijce.

Na koniec malutka niespodzianka-wklepywanka. Nie chcę cały czas zanudzać Czytelnika swoimi wywodami — proponuję wpisanie tego programu. Jego budową zajmiemy się później. Program ten kreśli na ekranie 16-kolorowego fraktala w rozdzielczości 640 na 512 punktów. Ciekawe jest to, że w żadnej dokumentacji od AMOSa nie spotkałem się z informacją jak otrzymywać tryby graficzne interlace (rozdzielczość pionowa 512 (PAL)), tak jakby AMOS ich w ogóle nie miał. Nikt nawet nie śmiał o tym napisać. Wstydzili się czy co? Tymczasem AMOS może oczywiście pracować w tych trybach czego dowodem jest poniższy program. Dokładniej opiszę ten problem podczas omawiania screenów. Jeśli ktoś ma kompilator to proponuję program ten skompilować, co przyspieszy jego wykonywanie około dwa razy. Wersję wynikową programu najlepiej będzie uruchomić przed wyjściem do kolegi, a do stacji df0: włożyć odbezpieczony dysk, na którym jest około 50 kilobajtów wolnego miejsca.

Kreślenie fraktala jest bardzo żmudne i długie, ale daje niezły efekt. Po zakończeniu rysunek zostanie zapisany na dysku pod nazwą „fractal.iff".

RAFAŁ BORZYŃSKI
(RABOCOST)

```
'*****************************************************
'*                  Fractal - AMOS                   *
'*                RABOCOST  05.1992                  *
'*         zgodny z  AMOS 1.1/1.2/1.3/Easy           *
'*****************************************************
'
'
'############# Przygotowanie screenu ############
'
Hide On
Screen Open 0,640,512,16,-1
Curs Off
Flash Off
Palette 0,$F,$F00,$FF,$F0F,$FF0,$FFF,$888,$8,$80,$800,$88
Palette ,,,,,,,,,,,,$880,$808,$F8F
Ink 0
Paint 10,10
'
'############# Definicje zmiennych ##############
'
MXORBITA=100
MNA#=-2
MXA#=1
MNB#=-1.5
MXB#=1.5
MXX#=640
MXY#=512
KROKA#=(MXA#-MNA#)/MXX#
KROKB#=(MXB#-MNB#)/MXY#
A#=MNA#
'
'############# Obliczenia i rysowanie ###########
'
For I#=1 To MXX#
   B#=MNB#
   For J#=1 To MXY#
      X#=0
      Y#=0
      For Q#=1 To MXORBITA
         V#=X#*X#-Y#*Y#+A#
         Y#=2*X#*Y#+B#
         X#=V#
         If X#<-2 or X#>2 or Y#<-2 or Y#>2 Then Goto BR
      Next Q#
      BR:
      If Q#=MXORBITA Then KOL=0 : Goto RET

      KOL=Q#-1
      RET:
      Ink KOL
      Plot I#,J#
      B#=B#+KROKB#
   Next J#
   A#=A#+KROKA#
Next I#
'
'############## Zgranie rysunku na dysk #########
'
Save Iff "df0:Fractal.IFF",1
End
```

TRACK BALL TKB-MT-AC

Upał nie pozwala się ruszyć z domu, można więc zasiąść do swojej przyjaciółki. Z pewną nieśmiałością sięgam po pudełko z napisem TRACK BALL.

Muszę przyznać, że "kot" (wydaje mi się, że kiedyś ktoś to tak nazwał... a może się mylę) jest bardzo estetycznie wykonany. Obudowa zrobiona jest z białomlecznego plastiku, dobrze harmonizującego z obudową Amigi. Kulka umieszczona niemal w centrum urządzenia jest przezroczysta i daje się łatwo obracać (nawet bez namoczenia). Całość zaopatrzono w trzy przyciski umieszczone odpowiednio na lewym i prawym boku, oraz na dole track balla, a także długi, bo około dwumetrowy przewód pozwalający na wygodne przyłączenie urządzenia do komputera.

KONSTRUKCJA

Track ball TKB-MT-AC jest urządzeniem opto-mechanicznym. Trwałość kuli przewidziana jest na „zrobienie" 100 mil, a norma trwałości klawisza to 1 milion przełączeń. Rozdzielczość track balla wynosi 162 punkty na cal. Urządzenie może pracować tak z Amigą jak i z komputerami serii Atari ST. Odpowiedni tryb pracy jest ustawiany za pomocą przełącznika.

DOKUMENTACJA, GWARANCJA I SERWIS

Firma Adenek udziela na sprzedawany wyrób dwuletniej gwarancji na ogólnie przyjętych zasadach. W kredowej, niebieskiej, sztywnej okładce znajduje się dwukartkowa instrukcja obsługi składająca się z 16 zdań w czterech językach świata i oczywiście bez języka polskiego. O instrukcji wystarczy powiedzieć jedynie, że jest po prostu denna. Obok instrukcji w pudełku znajduje się także ładnie wydany, malutki katalog innych wyrobów firmy Alfa Data.

W AKCJI

Zaraz po włączeniu zasilania dostrzegłem kątem oka błysk w kulce track balla. Zaniepokojony przytknąłem do niego nos, ale nic szczególnego nie wywąchałem. Po chwili jednak odkryłem coś, co bardzo mi się spodobało: otóż podczas przyciskania klawiszy track balla, kula świeci różnymi kolorami.

Na początku nie mogłem się przyzwyczaić



do tej nowości. Często chcąc przesunąć na ekranie wskaźnik, łapałem całe urządzenie, próbując przesuwać je po stole. Najlepiej jednak zachował się mój kolega, który dobrał się do komputera podczas mojej chwilowej nieobecności w pokoju. Gdy wróciłem, ujrzałem jak spocony, cicho szepczący coś pod nosem, stara się używać „kota" jak zwykłą, szarą myszkę. Mimo jego najszczerszych starań track ball także nie dał się zepsuć.

Już po kilku godzinach pracy z „kotem" przekonałem się o jego niepodważalnej przewadze nad myszą (zwłaszcza tą sprzedawaną w komplecie z Amigą). Po pierwsze, jako urządzenie statyczne, track ball zajmuje bardzo mało miejsca na stole, nie trzeba również używać podkładki pod mysz. Minimalnymi ruchami czubków palców można osiągnąć to, co myszą można było osiągnąć dopiero po dużym ruchu całej ręki. Jest to bardzo ważne, zwłaszcza przy wielogodzinnej pracy z Amigą.

Bardzo wygodny jest także dodatkowy, trzeci klawisz track balla. Jego jednorazowe wciśnięcie odpowiada normalnemu przytrzymywaniu "tradycyjnego" — lewego. Korzyści wynikające z jego umieszczenia są ogromne — wystarczy wczytać jakikolwiek program graficzny, nie mówiąc już o grach. Konstrukcja i rozmieszczenie kuli oraz klawiszy w obudowie wydają mi się przemyślane pod każdym względem. Należy także dodać, że klawisze (przyciski) wciska się bardzo lekko, co dodatkowo zwiększa komfort pracy.

Po ogólnym zapoznaniu się z urządzeniem postanowiłem przejść do cięższych technik. Do stacji powędrował stary, dobry hit — strzelanina „Battle Squadron". Kto widział tę grę, wie, że jest ona jedną z najcięższych prób dla myszy i joysticka. Specjalnie wynajęta ekipa młodszych kolegów przystąpiła do akcji. Track ball nie poddał się, nie "wysiadł" również podczas wielogodzinnej męczarni (brydż, pasjanse, tetrix), którą sprawił mu mój ojciec, mający bardzo ciężką rękę. Ogólnie można powiedzieć, że testowany przeze mnie "kotek" wydaje się być zwierzęciem bardzo wytrzymałym.

NASZ WERDYKT

Podsumowując mogę śmiało stwierdzić, że track ball TKB-MT-AC jest wyrobem dopracowanym pod względem estetycznym i funkcjonalnym. Polecałbym go wszystkim, którzy korzystają ze standardowych amigowskich myszy. Różnica w komforcie pracy jest ogromna i nieporównywalna.

RAFAŁ BORZYŃSKI
(RABOCOST)

DYSTRYBUTOR:
ADENEK, Raszyn, ul. Mickiewicza 14, tel. 56-08-91.

DANE TECHNICZNE:

Rozdzielczość	: 162 DPI
Wymiary	: 140 x 105 x 156 mm
Waga	: 300 g
Tryby pracy	: Amiga lub Atari ST

ZALETY:

1. Wysoka estetyka i jakość wykonania (bardzo estetyczny jest efekt świecenia kuli)
2. Mała siła potrzebna do obracania kuli i wciskania klawiszy
3. Uniwersalność zastosowania (Atari ST/Amiga)
4. Bezawaryjność pracy i duża wytrzymałość mechaniczna
5. Długi przewód połączeniowy pozwalający na wygodne umieszczenie track balla
6. Uzasadnione i wygodne rozmieszczenie części ruchomych

WADY:

1. Brak instrukcji w języku polskim (chociaż trudno mówić o przydatności instrukcji w przypadku tego urządzenia)



ULEPSZONY SAMPLER DLA AMIGI 500 i 2000

W tym artykule przedstawiam kolejny sampler dla Amigi. Tym razem jest to profesjonalny sampler o jakości Hi-Fi, lecz jednocześnie dość kosztowny. Koszt wykonania może przekroczyć 500 tys. zł. Na wstępie należy także uprzedzić potencjalnych konstruktorów, że układ przetwornika a/c zastosowanego w układzie jest trudnodostępny.

Opisana poniżej konstrukcja charakteryzuje się czasem konwersji 15 mikrosekund, co daje częstotliwość próbkowania rzędu 70 kHz. Możliwe jest więc uzyskanie stereofonicznego samplingu z częstotliwością 35 kHz co odpowiada pasmu przenoszenia do 17 kHz, a więc mieszczącemu się spokojnie w normach Hi-Fi. Jeżeli dodać do tego bardzo niskie szumy własne konwertera (przetwornika) a/c to otrzymujemy pełnowartościowy stereofoniczny sampler wysokiej jakości.

Układ należy zmontować na płytce drukowanej, której schemat zamieszczono na rysunku obok. Po sprawdzeniu poprawności montażu układ należy podłączyć do portu równoległego (Parallel Port) i jednego z wyprowadzeń portu szeregowego (Serial Port). Z tego drugiego portu jest pobierane ujemne napięcie odniesienia czyli —12V.

Teraz należy wczytać program Audiomaster, następnie wybieramy z menu Project opcję MONITOR. Widniejąca w oknie pozioma kreska powinna być mniej więcej w środku tego okna. Jeżeli tak nie jest, należy to skorygować zmieniając ustawienie potencjometru P1. Teraz możemy spróbować dokonać samplingu. Jeśli dźwięk jest zniekształcony, to może to oznaczać uszkodzenie przetwornika lub błąd w układzie. Jeżeli zdecydowaliśmy się na wersję stereofoniczną (UWAGA! PŁYTKA DRUKOWANA PRZEDSTAWIONA NA RYSUNKU ZAPROJEKTOWANA ZOSTAŁA DLA WERSJI MONO), a otrzymany dźwięk jest monofoniczny (słychać jeden kanał bądź oba, ale monofonicznie) to jest to oznaką uszkodzenia układu kluczującego 4066. W tym przypadku należy wymienić go na inny egzemplarz, gdyż często spotyka się w tym układzie zjawisko „zatrzaskiwania" się kluczy (ang. latch-up).

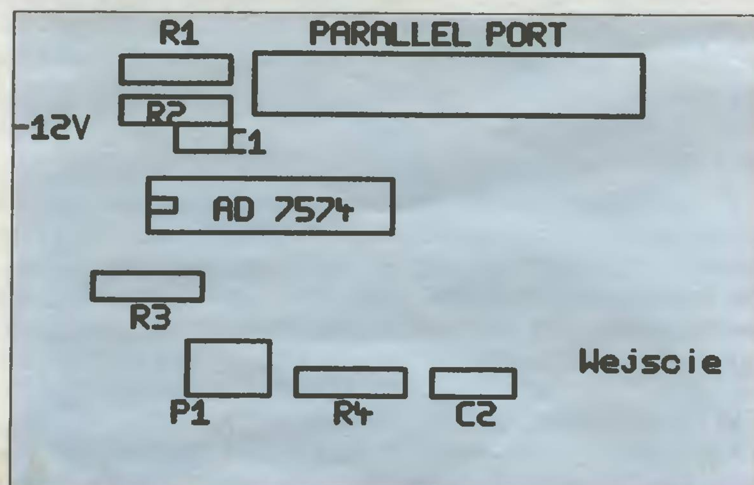
Parę słów na temat samego przetwornika. Jest to układ produkowany przez firmę Analog Devices. Jest on podobny (funkcjonalnie) do przetwornika zastosowanego w poprzednim samplerze, który opisywałem już na łamach "C&A". Układ zastosowany w nowym samplerze jest jednak od niego szybszy i może być taktowany zegarem wewnętrznym (z tej możliwości skorzystano w opisywanym samplerze). Sterowanie układem zapewniają sygnały: CS — Chip Select, RD — Read/Write, BUSY — sygnał oznaczający konwersję w toku. Zakres napięć wejściowych (analogowych) powinien zawierać się w zakresie ±20 V.

Co do jakości jaką pozwala uzyskać opisany tu sampler odsyłam Czytelników do specjalnej dyskietki dla Amigi wydanej wraz z numerem „C&A" nr 06/02. Zamieszczony tam sampling grupy Dire Straits jest moim zdaniem wystarczająco wymowny. Schemat ideowy przedstawiam na str. 32.

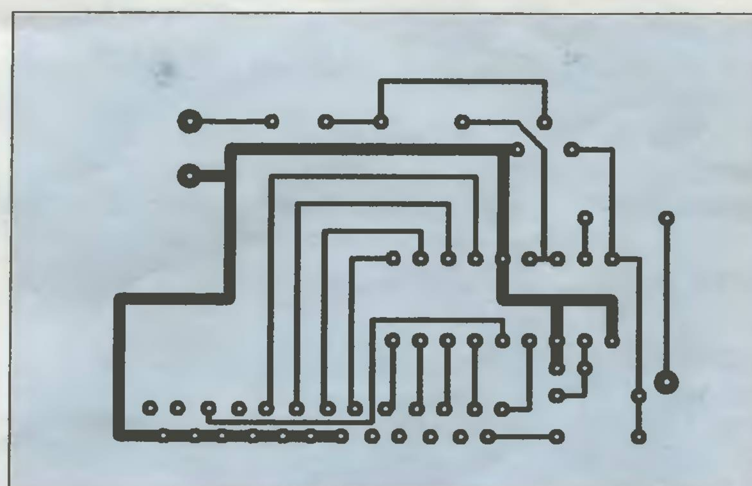
Dużo zadowolenia z wykonanego samplera życzy

JERZY "BLUE THUNDER" DUDEK

NOWY SAMPLER



Rozmieszczenie elementów



Płytka

6526 COMPLEX INTERFACE DEVICE # 2

Spotykamy się już po raz trzeci. Nadszedł czas na deser. Ponieważ układy CIA#1 i CIA#2 są identyczne pominę szczegółowy opis rejestrów CIA#2, gdyż większość jest po prostu taka sama.

Na początek bardzo ważna uwaga. Układ CIA#1 ma dołączoną linię przerwań do linii IRQ procesora (przerwania maskowalne), natomiast linia przerwań CIA#2 jest dołączona do linii NMI procesora (przerwań tych **NIE MOŻNA WYŁĄCZYĆ**). Jak widać, niektórzy pracują więcej, inni mniej. Odnosi się to także do układów CIA#1 i 2. „Jedynka" pracowicie obsługuje klawiaturę i joysticki, a „dwójka" — wszelkie operacje dyskowe i transmisję danych poprzez port szeregowy oraz RS-232. Innym zadanie CIA 2 jest przydzielanie banków pamięci dla układu VIC (4 banki po 16 KB). Zaczynamy.

54576 ($DD00) CI2PRA CIA#2+0

Port danych A.

bity 0—1: wybór banku pamięci dla układu VIC;
- 11 — bank 0 ($0000—$3FFF);
- 10 — bank 1 ($4000—$7FFF);
- 01 — bank 2 ($8000—$BFFF);
- 00 — bank 3 ($C000—$FFFF);

bit 2: RS-232 — wyjście danych (złącze M USER PORT);
bit 3: SERIAL PORT — sygnał wyjściowy ATN;
bit 4: SERIAL PORT — zegar nadajnika (wyjście — impuls taktujący);
bit 5: SERIAL PORT — wyjście danych;
bit 6: SERIAL PORT — zegar odbiornika (wejście — impuls taktujący);
bit 7: SERIAL PORT — wejście danych.

UWAGA:

***SERIAL PORT** — port szeregowy (w wypadku C-64/128 także szyna systemowa).*

***USER PORT** — port użytkownika zawierający m.in. linie złącza RS-232.*

54577 ($DD01) CI2PRB CIA#2+1

Port danych B

bit 0: RS-232 — wejście danych (SIN) — złącze C USER PORT;
bit 1: RS-232 — RTS — złącze D USER PORT;
bit 2: RS-232 — DTR — złącze E USER PORT;
bit 3: RS-232 — RI — złącze F USER PORT;
bit 4: RS-232 — DCD — złącze H USER PORT;
bit 5: złącze J USER PORT;
bit 6: RS-232 — CTS — złącze K USER PORT;
: wyjście sygnału timera A;
bit 7: RS-232 — DSR — złącze L USER PORT;
: wyjście sygnału timera B;

Myślę, że czeka mnie teraz masa tłumaczenia. Dwa pierwty rejestru CI2PRA określają, który z czterech 16. KB banków ma być przydzielony dla układu VIC (przestrzeń adresowa VIC wynosi 16 KB). W jednym takim banku muszą mieścić się dane dla duszków, pamięć ekranu graficznego i tekstowego, zestaw znaków. Standardowo VIC korzysta z banku 0 ($0000 — $3FFF).

W celu zmiany banku należy ustawić bity 0 i 1 CI2PRA jako wyjście (metoda ta była opisywana przy układzie CIA#1):

```
POKE 56578, PEEK(56578)OR3
```

oraz wybrać bank:

```
POKE 56576, (PEEK (56576) AND252) OR(3—X)
```

gdzie X — numer banku (0—3).

W banku zerowym nie ma zbyt wiele miejsca na dane graficzne. Tutaj znajduje się strona zerowa pamięci o kluczowym znaczeniu dla systemu. Tutaj też zlokalizowana jest pamięć ekranowa, pamięć przeznaczona na programy w BASIC. Ponadto właśnie w tym banku (4096—8192) VIC „widzi" generator znaków! W rzeczywistości wolnego miejsca prawie tutaj nie ma jeśli nie liczyć komórek 679—767 i 820—1023 (np. na 4 kształty duszków o numerach 11, 13, 14, 15). Bufor magnetofonu pomijam.

W banku 1 jest znacznie luźniej, bo w tym miejscu znajduje się tylko pamięć przeznaczona na programy w BASIC. Przy krótkich programach (ok. 12 KB) można bez większego problemu wykorzystać ten obszar. Jednak w tym obszarze VIC nie „widzi" generatora znaków. Wymaga to więc skopiowania całego zestawu znaków do tego obszaru, co nie jest zbyt wygodne.

W banku 2 znajduje się 8 KB ROM generatora znaków, oraz 8 KB ROM interpretera BASIC. Nie znaczy to jednak, że obszaru tego nie można wykorzystać. Przecież pod ROM znajduje się pamięć RAM i można ją bez problemów wykorzystać; wymaga to jednak znajomości asemblera.

Komórki $C000—$CFFF to obszar często wykorzystywany przez różne programy w banku 3. Znajduje się tu także 8 KB ROM KERNAL (system operacyjny). Tutaj jest też newralgiczny obszar adresowy układów SID, VIC, CIA. Zalecana jest więc ostrożność.

Bit 5 CI2PRA to wyjście danych, a 7 to wejście danych do i z portu szeregowego. Bity 4 i 6 to impulsy zegara traktujące odpowiednio nadawanie i odbieranie danych przesyłanych poprzez port szeregowy. Bit 3 — przesłanie sygnału ATN na serial port.

Bardzo ważną zaletą (aczkolwiek prawie niewykorzystywaną) jest zapisana w pamięci ROM C-64 obsługa złącza RS-232. Wytłumaczenie działania tego złącza wymagałoby niestety przepisania połowy ROM komputera. W tym miejscu chciałbym odesłać wszystkich zainteresowanych do mapy pamięci zawartej w książce „Jak rozbudować interpreter" B. Radziszewskiego i K. Gajewskiego.

Sygnały na bitach 7—6 i 4—1 CI2PRB są standardowymi sygnałami sterującymi pracą złącza RS-232:

bit 7 — DSR — Data Set Ready;
bit 6 — CTS — Clear To Send;
bit 4 — DCD — Data Carrier Detect;
bit 3 — RI — Ring Indicator;
bit 2 — DTR — Data Terminal Ready;
bit 1 — RTS — Request To Send;

UWAGA:
Jedną z różnic w standardzie złącza RS-232 stosowanego np. w Amidze i w C-64 są napięcia — w Commodore 64/128 są one ograniczone do poziomu napięć TTL, co wymaga konwersji napięć. Maksymalną prędkość transmisji jaką można w wypadku C-64 uzyskać jest 2400 bodów — szybszej transmisji nie jest w stanie obsłużyć oprogramowanie systemowe C-64 (staje się ono zbyt wolne).

Bit 2 CI2PRA jest używany jako wyjście danych nadawanych. Bit 0 CI2PRB — to wejście odbieranych danych. Tabela 1 przedstawia rozkład wyprowadzeń złącza USER PORT (portu użytkownika).

56578 ($DD02) C2DDRA CIA#2+2
Rejestr kierunku danych portu A.
Opisałem już działanie tych rejestrów w CIA#1. Tutaj działają tak samo. Standardowo C2DDRA ustawiany jest na wartość 63 (bity 6, 7 — wejście, pozostałe — wyjście).

56579 ($DD03) C2DDRB CIA#2+3
Rejestr kierunku danych portu B.
Standardowo jest ustawiany na 0 (wszystkie bity jako wejście). Bity 1 i 2 zmieniane są na wyjście w chwili korzystania z RS--232.
Dalej następują 4 rejestry timerów A i B. Działają dokładnie tak samo jak w wypadku CIA#1. Są używane do taktowania transmisji danych poprzez RS-232. Timer B jest także wykorzystywany do obsługi transmisji poprzez port szeregowy.

56580 ($DD04) TI2ALO CIA#2+4
Młodszy bajt timera A.

56581 ($DD05) TI2AHI CIA#2+5
Starszy bajt timera A.

56582 ($DD06) TI2BLO CIA#2+6
Młodszy bajt timera B.

56583 ($DD07) TI2BLO CIA#2+7
Starszy bajt timera B.

Cztery kolejne rejestry to zegar czasu rzeczywistego (dokładnie taki sam jak w CIA#1). System operacyjny nie wykorzystuje tego zegara.

56584 ($DD08) TO2TEN CIA#2+8
Rejstr dziesiętnych sekundy zegara czasu rzeczywistego.

56585 ($DD09) TO2SEC CIA#2+9
Rejestr sekund zegara czasu rzeczywistego.

56586 ($DD0A) TO2MIN CIA#2+10
Rejestr minut zegara czasu rzeczywistego.

56587 ($DD0B) TO2HRS CIA#2+11
Rejestr godzin zegara czasu rzeczywistego z wskazaniem przed/po południu.

56588 ($DD0C) CI2SDR CIA#2+12
Port szeregowy.
Taki sam port szeregowy znajduje się w CIA#1. System operacyjny nie korzysta z tego portu.

56589 ($DD0D) CI2ICR CIA#2+13
Rejestr sterowania przerwaniami.
Działa analogicznie jak w wypadku CIA#1. Linia flag CIA#2 dołączona jest do złącza B USER PORT. Umożliwia to generowanie przerwań NMI przez źródło zewnętrzne.

56590 ($DD0E) CI2DRA CIA#2+14
Rejestr sterujący A.
Dokładny rozpis tego rejestru znajdziesz w poprzednim odcinku. Jedyna różnica to źródło impulsów dla timera A, które można dołączyć do linii CNT2 (złącze 6 USER PORT).

56591 ($DD0F) CI2CRB CIA#2+15
Rejestr sterujący B.
Taki sam rejestr jak w CIA#1. Jedyną różnicą jest możliwość wykorzystania linii CNT2 (złącze 6 USER PORT) do taktowania timera B lub testowania stanu tej linii.

Obszar 56592—56931 ($DD10—$DDFF)
Zwierciadlane odbicia rejestrów CIA#2.

Układ CIA#2 ma tylko 16 rejestrów. Jego przestrzeń adresowa to 256 bajtów. Dekodowane są tylko 4 najmłodsze bity adresowe. W efekcie 2 256-bajtowym bloku każdy 16 obszar jest nierozróżnialny dla systemu. Tak więc 56576+K*16 (K od 0 do 15) to ten sam rejestr.

Pozostaje jeszcze do omówienia cała transmisja danych poprzez złącze RS-232 czym zajmiemy się niebawem.

BARTŁOMIEJ DRAMCZYK

ASEMBLER 6502

czyli adresowanie indeksowane oraz pośrednie i co z tego wynika

Czym właściwie jest adresowanie? Zadaję to pytanie nie bez powodu, chociaż, na dobrą sprawę, bez dokładnej znajomości tego terminu można się doskonale obejść. Jednak jasne i jednoznaczne wyłożenie tej kwestii może być zwyczajnie dużym uproszczeniem w tłumaczeniu różnych spraw z programowaniem związanych. O trybach nie wspominałem na początku naszych spotkań po prostu dlatego, że nie wydawało się to konieczne. Teraz jednak tłumaczenie coraz bardziej zawiłych problemów bez wprowadzenia tej terminologii staje się coraz trudniejsze. Dlatego też postanowiłem poświęcić dzisiejszy artykuł właśnie trybom adresowania.

Zacznijmy więc jeszcze raz: Czym właściwie jest adresowanie? Adresowanie to sposób podania komputerowi informacji o tym, jak każdy rozkaz ma być wykonany. Jeżeli dalibyśmy rozkaz np. LDA 39, to niekoniecznie byłoby wiadomo, o co nam właściwie chodzi. Dlatego też zostały opracowane różne tryby adresowania. Jeżeli przed liczbą (argumentem) postawimy znak "#", to jako argument będzie pobrana liczba, którą wpisaliśmy za rozkazem. Ten tryb adresowania nazywamy absolutnym lub natychmiastowym. Na przykład jeśli wydamy rozkaz LDA # $98 to w akumulatorze znajdzie się szesnastkowa liczba 98.

Kolejnym prostym trybem adresowania jest adresowanie bezpośrednie (ang. immediate). Podana po rozkazie liczba jest przez komputer traktowana jako adres, pod którym ma szukać argumentu. Jeśli więc napiszemy LDA $3000, komputer odszuka w pamięci komórkę o adresie szesnastkowym 3000 (dziesiętnie 12288) a następnie jej zawartość wczyta do akumulatora. Analogicznie dzieje się, kiedy napiszemy STA $1000. Wtedy zawartość akumulatora zostanie wpisana do komórki o adresie $1000 (4096).

Szczególną odmianą tego adresowania jest adresowanie strony zerowej, czyli pierwszych 256 komórek pamięci. Jej odmienność polega na tym, że do adresowania strony zerowej potrzebna jest tylko jedna komórka. Właśnie dlatego strona zerowa jest tak często przez programistów wykorzystywana — daje to czasem duże oszczędności czasowe.

Innym trybem adresowania jest adresowanie pośrednie. Jego idea polega na tym, że adres podany po rozkazie JMP (bo JMP jest JEDYNYM rozkazem, który może być tak adresowany) jest dla komputera adresem pod którym znajduje się adres (w postaci młodszego i starszego bajtu) procedury, do której należy skoczyć. Wygląda to np. tak: JMP ($0314). Pod adresem znajduje się wektor, który wskazuje na komórkę, do której trzeba skoczyć (zwykle w tym przypadku $ea31).

Jakieś dwa miesiące temu rozmawialiśmy trochę o tym, jak wydrukować na ekranie napisy. Robiliśmy to jeszcze bez użycia różnych procedur zapisanych w ROM komputera. Jak zapewne pamiętacie, używaliśmy do tego celu rejestru X jako rejestru pomocniczego. Właśnie operacje w rodzaju LDA $AAAA,X albo STA $BBBB,X nazywamy operacjami indeksowanymi. Zaś takie podanie instrukcji LDA lub np. STA nazywane jest adresowaniem indeksowanym. Indeksować można też oczywiście i inne rozkazy, ale o tym — potem. Przypomnę jeszcze tylko znaczenie takiego adresowania. Rozkaz LDA 10000,X oznacza dla komputera, że do akumulatora ma wstawić liczbę, która znajduje się w komórce o adresie obliczonym przez dodanie 10000 do zawartości rejestru X. Rozkazy LDA i STA można również indeksować drugim z rejestrów pomocniczych, Y.

Dwa kolejne tryby adresowania nie są już takie oczywiste. Tryby te to pośredni postindeksowany i pośredni preindeksowany. Adresowanie postindeksowane ma składnię LDA ($fb),y. Liczba podana po rozkazie jest tu miejscem, w którym znajduje się adres w pamięci. Do adresu tego trzeba dodać jeszcze zawartość rejestru Y i dopiero ta liczba wskazuje lokację pamięci, z której należy pobrać liczbę do akumulatora. Do tego indeksowania można zastosować wyłącznie rejestr Y. Adresowanie preindeksowane polega na tym, do liczby po rozkazie dodajemy jeszcze zawartość rejestru X i ta suma jest adresem komórki, której zawartość wstawiamy do akumulatora. Składnia wygląda tak: LDA ($f0, X) i trzeba pamiętać, że do indeksowania można użyć wyłącznie rejestru X. Moim skromnym zdaniem, tryb ten jest dość mało praktyczny.

W rozkazach skoków warunkowych, jak np. poznane już przez nas BEQ i BNE, mimo tego, że podajemy pełne adresy pod które należy skoczyć, komputer przechowuje tylko liczbę komórek o którą musi przeskoczyć w tył lub w przód. Dlatego też nazywamy ten tryb adresowaniem względnym.

Dwa ostatnie tryby, które zostały nam do omówienia są stricte wewnętrzne i użytkownik ma na nie znikomy wpływ. Pierwszy to adresowanie akumulatora i obejmuje wszystkie operacje wykonywane na samym akumulatorze, bez udziału pamięci. Drugi zaś — to adresowanie niejawne. Należą do niego podobne rozkazy, np. poznany przez nas INX. Zwiększa on zawartość rejestru X o 1 i wcale nie interesuje nas, gdzie we wnętrzu komputera znajduje się komórka, której rozkaz ten dotyczy. Do rodzinki tej należą jeszcze NOP, RTS, BRK i inne tego typu rozkazy.

Wiem, że podanie wszystkich trybów adresowania było troszkę „na wyrost", ale na pewno przyda się później. Przepraszam, że może byłem tym razem trochę nudnawy i nie podałem listingów przykładowych, ale taka solidna dawka teorii zaprocentuje na pewno w przyszłości.

BARTEK KACHNIARZ

ZABAWY ZE STOSEM

czyli samouruchamianie się po raz drugi

W niniejszym tekście chciałbym przedstawić zapowiedziane poprzednio samouruchamianie realizowane przy użyciu stosu mikroprocesora. Mała uwaga techniczna: będę operował liczbami szesnastkowymi. Wprawdzie nieobeznanym może się to wydać niewygodne, ale zapewniam wszystkich, że jeśli ich trochę poużywają, to nie będą się mogli bez niego obejść. Spróbujcie np. obliczyć młodszy i starszy bajt liczby dziesiętnej i szesnastkowej (heksadecymalnej). Różnica jest chyba widoczna od razu...

Na początek trochę podstawowych wiadomości o samym stosie. Nie będę się tu bardzo rozwodził, ale ogólne pojęcie jednak trzeba mieć.

Stos jest to pewien obszar pamięci (ściśle określony) przeznaczony dla mikroprocesora i systemu m.in. do przechowywania tzw. adresu powrotnego i podprogramu i obsługi przerwań, oraz do przechowywania zawartości rejestrów wewnętrznych mikroprocesora. Ze stosu korzysta również interpreter podczas wykonywania pętli FOR-NEXT, instrukcji GOSUB i DEF.

Stos zajmuje on obszar od $0100 do $01ff (czyli całą pierwszą stronę pamięci). Nazwa „stos" charakteryzuje najlepiej organizację tej pamięci: tu możemy jedynie zapisać liczbę, lub ją zdjąć. Ważne jest przy tym, że dostępna bezpośrednio jest tylko ostatnio zapisana na stosie liczba, kolejne zaś — dopiero po „zdjęciu" ostatniej wpisanej. Technika ta nazywa się LIFO (Last In, First Out).

Stos kontrolowany jest przez jeden z rejestrów mikroprocesora, tzw. wskaźnik stosu. Jego zadaniem jest określenie ostatnio użytej lokalizacji. Stos zapełnia się od góry (od komórki $01ff) w dół (do $0100). Dozwolone operacje to **PHA** — zapis akumulatora na stos, **PLA** — zdjęcie akumulatora ze stosu, **PHP** — zapis rejestru słowa stanu (znaczniki) i **PLP** — odczyt („zdjęcie") rejestru słowa stanu. My wykorzystamy jedynie instrukcję PHA.

Teraz proponuję małe doświadczenie, które powinno wyjaśnić całą ideę samouruchamiania. Jak już wspomniałem, na stosie przechowywane są adresy powrotne z podprogramów, a to jest właśnie to, czego nam potrzeba. Wystarczy więc za pomocą **PHA** zapisać tu adres początkowy naszego programu zmniejszony o jeden, w postaci starszego i młodszego bajtu i wykonać rozkaz RTS. Proszę zwrócić uwagę na kolejność: najpierw starszy, potem dopiero młodszy bajt. Nie jest to pomyłka, ponieważ zgodnie z tym co napisałem wcześniej stos zapełniany jest w dół (od $01ff) i w pamięci wszystko będzie w porządku. Najlepiej zobaczyć to na przykładzie.

```
$1000 LDA #$2f
      PHA
      LDA #$ff
      PHA
      RTS
```

Teraz od adresu $3000 umieszczamy cokolwiek, byle efekt był widoczny, np.:

```
LDA #$00; kod koloru czarnego
STA $d021; do rejestru koloru tła
RTS
```

Wykonujemy instrukcję monitora G1000 lub odpowiadającą i widać od razu co się stało...

Teraz pozostaje już tylko w dolną część stosu wpisać tzw. "loader" (jak go stworzyć pisałem poprzednio), czyli procedurę odpowiedzialną za wczytanie programu lub kolejnej jego części. Resztę wolnego miejsca zapełnij adresem początkowym, pomniejszonym o jeden. Z oczywistych względów liczba ta powinna mieć młodszy i starszy bajt identyczny, dlatego też proponuję sam "loader" umieścić od adresu $0102, a wolną część stosu wypełnić wartością $01. Tutaj jednak napotykamy na mały problem. Program ten nie bardzo można w całości układać w obszarze, do którego będzie potem przeniesiony. Dlaczego?

Jeśli za pomocą monitora języka maszynowego wczytamy program wykorzystujący przy samouruchamianiu wektor pętli głównej interpretera BASIC, nic się nie stanie. Taki program uruchomi się dopiero w momencie zakończenia pracy z monitorem. Z kolei przy wykorzystaniu stosu, ta metoda niestety nic nie da, ponieważ program natychmiast się wykona. Można zresztą zrobić doświadczenie, jeśli ktoś nie wierzy na słowo. Aby ominąć te niedogodności napiszemy program, który utworzy na dysku żądany plik.

```
; LISTING 1
;
;
LDX #$08
LDX #$01     — parametry pliku
TAY
JSR $FFBA
LDA #$04
LDX #$00     — nazwa pliku
LDY #$C0
JSR $FFBD
JSR $FFC0    — otwórz plik
LDX #$01
JSR $FFC9
```

```
      ; LISTING 2
      ;
      ;
      LDY #$00  —
loop  LDA $0100,Y — pętla przesyłająca treść programu
      JSR $FFD2     loadera + 2 pierwsze bajty jako
      INY           adres ładowania ($0102)
      CPY #liczba bajtów tworzących loader
      BNE loop   —
      LDA #$01   —
loop2 JSR $FFD2  — dopełnianie stosu wartością $01
      INY
      BNE loop2  —
      JSR $FFC3  — zamknięcie pliku. Pamiętaj, że
      JSR $FFCc    w akumulatorze jest wartość $01.
      RTS          Ekran i klawiatura to aktualne
                   urządzenia wejścia/wyjścia oraz
                   powrót do BASIC.
```

Wbrew pozorom nie jest to takie trudne. Wystarczy nam do tego kilka nowych procedur (część opisałem poprzednio), które zaraz przedstawię.

Procedura OPEN

Dostępna bezpośrednio pod adresem $f34a, poprzez tablicę skoków Kernal $ffc0 i wektor $031a. Za jej pomocą otwieramy plik oraz tzw. kanał łączności z danym urządzeniem peryferyjnym (w naszym przypadku ze stacją dysków). Innymi słowy pozwala ona na przesyłanie danych między komputerem, a urządzeniem peryferyjnym.

Procedura CHKOUT

Dostępna bezpośrednio przez $f2540, tablicę skoków $ffc9 i wektor $0320. Pozwala ona wysyłać dane na inne urządzenie niż ekran. Należy ją wykonać zaraz po OPEN. Przed wywołaniem do rejestru .X wpisujemy numer pliku logicznego określony wcześniej za pomocą procedury SETLFS). Po wywołaniu plik logiczny (z .X) staje się plikiem aktualnym (tym, którego będziemy używać w następnej kolejności). Podobnie określone jest urządzenie i adres pomocniczy.

Procedura CLOSE

Dostęp: tablica skoków ffc3, bezpośrednio $f291, wektor $031c. Używamy jej do zamknięcia pliku i kanału łączności po wykonaniu wszystkich operacji wejścia/wyjścia. Przed wywołaniem należy wpisać do akumulatora numer używanego pliku logicznego. Warto pamiętać, że wywołanie jej jest konieczne, aby dany plik mógł być w ogóle odczytany.

Procedura CHROUT

Dostęp: tablica skoków $ffd2, bezpośrednio $f1ca, wektor #0326. Wysyła znak z akumulatora do aktualnego urządzenia wyjściowego. Jeżeli nie użyjemy wcześniej OPEN i CHKOUT, to znak zostanie wyświetlony na ekranie. Akceptuje ona również kody sterujące, np. $93 (147) — czyszczenie ekranu. Jeżeli urządzeniem wyjściowym jest magnetofon, to znak z akumulatora jest wpisywany do 192. znakowego bufora i dopóki nie zostanie on zapełniony, rzeczywista transmisja nie ma miejsca.

Procedura CLRCHN

Dostęp: tablica skoków $ffcc, bezpośrednio $f333, wektor $0322. Ustawia ona klawiaturę jako aktualne urządzenie wejściowe, a monitor jako wyjściowe, czyli przywraca stan pierwotny. Dla zainteresowanych — jeżeli urządzeniem wejściowym lub wyjściowym było urządzenie szeregowe, to na szynę szeregową wysyłane są odpowiednio polecenia UNTALK i UNLISTEN.

Teraz już sam progam, który tworzy na dyskietce plik o nazwie BOOT (listing 1). Jego zawartością będzie 256 bajtów, a ładować się i uruchamiać będzie od adresu $0102.

W tej chwili jesteśmy już gotowi do wysyłania danych. Proszę jednak pamiętać, że dwa pierwsze bajty pliku stanowią adres ładowania i trzeba je również uwzględnić. Program przedstawiono na listingu 2.

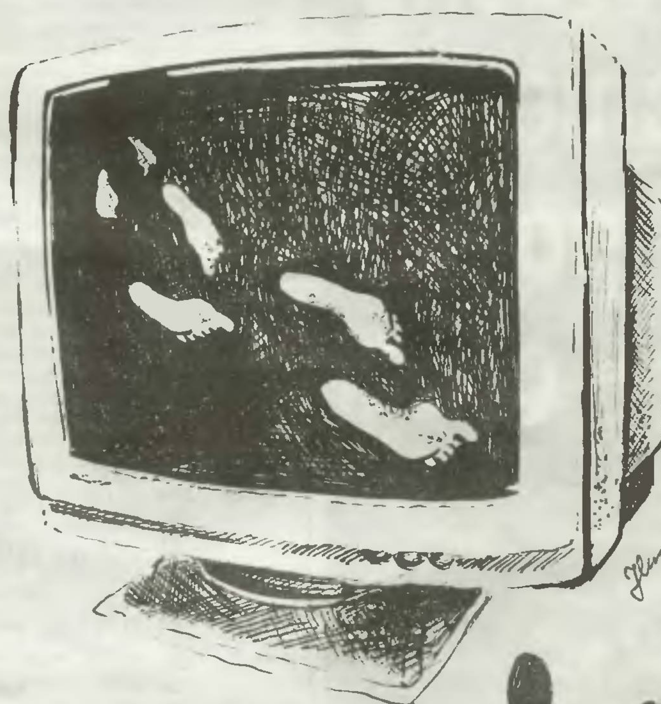
Na koniec jeszcze drobne wyjaśnienia. Od adresu $c000 musimy umieścić nazwę naszego programu, np. $42, $4f, $4f, $54. Sam loader wpisujemy na dół stosu od adresu $0102 począwszy. Jeśli nie będzie zbyt długi, to wszystko jest OK.

Jeszcze jedno. Istnieje też inna metoda realizacji tego zagadnienia, ale ta wydaje mi się znacznie ciekawsza. Teraz Czytelnik może tworzyć własne pliki z dowolną właściwie zawartością, wczytujące się pod żądany adres. Można też przesyłać dane z klawiatury bezpośrednio na dysk (jeżeli komuś będzie to potrzebne) i wiele innych ciekawych rzeczy. Zależy to już wyłącznie od pomysłowości Czytelników.

MARCIN "FOX" LIS

Pamiętnik znaleziony w armacie:

JAK WYJŚĆ Z RAMEK BOCZNYCH?



Kanonierzy, jak wszyscy, miewają wakacje. Dlatego też w numerze sierpniowym nie było kolejnego odcinka pamiętnika. Tym razem odpłacimy to Wam z nawiązką, więc... odkurzcie działa, klawiatury i monitory. Joystick można schować. Do działa... to jest — Do dzieła!

W lipcu kanonierzy mieli okazję, by po raz pierwszy zrobić coś — teoretycznie — niemożliwego. Opublikowaliśmy bowiem dwa równoległe, niezależne artykuły o otwieraniu dolnej i górnej obwódki. Przy okazji udało mi się obiecać, że napiszę o otwieraniu obwódek bocznych. Obiecałem to też kilku ludziom, którzy w tej sprawie pisali listy do redakcji. Zaczynamy.

Otwarcie obwódki dolnej było stosunkowo proste. Musieliśmy tylko dojść do którejś tam linii rastra, zwęzić ekran o 1 linię, a później — rozszerzyć go z powrotem. Generalnie można powiedzieć, że otwieranie ramki bocznej oparte jest na identycznej zasadzie. Szkopuł polega jednak na tym, że aby tego dokonać, trzeba zwęzić ekran, kiedy wiązka elektronów będzie celować w kolumnę o numerze 39. Rozszerzyć najlepiej będzie od razu potem. Żeby nie było nam zbyt prosto, operację tę musimy powtarzać w każdej linijce. Udało mi się ustalić, że w czasie, w którym procesor wykonuje jeden cykl, działo przesuwa się w poziomie o jedną kratkę (czyli osiem punkcików zwanych też pikselami). Wynika z tego, że musimy działać z ogromną precyzją, a ważne dla nas będą dosłownie milionowe części sekundy. Dlatego też — zwłaszcza z początku — będziemy popełniać sporo błędów i nasze programy będą wymagały stosunkowo częstego usuwania błędów. Spowodowało to również zmianę programu, z którego będziemy korzystać przy pisaniu programów — powinien to być TurboAssembler 5.1 (takiego właśnie używam).

Przejdźmy jednak do konkretów. Jeżeli chcemy otworzyć ramkę to najpierw musimy dokładnie określić, co będzie się znajdować na ekranie na wysokości, na której chcemy to zrobić. Musimy zdecydować, czy grafika będzie jedno- czy wielokolorowa, czy tryb będzie graficzny, czy tekstowy, ilu duszków będziemy używać itd. Ponadto ważne jest określenie, które to będą duszki.

Duszka o numerze 0 NIE UŻYWAMY. Bowiem każdy z włączonych duszków zabiera komputerowi dwa cykle, zaś duszek zerowy — aż pięć i to akurat w momencie, w którym mamy otwierać obwódkę. Jeśli jesteśmy już przy sprajtach — pamiętajmy, że znaczenie mają tylko te duszki, które są właśnie włączone i to niezależnie od tego, jak wyglądają. Sprajty „puste" też zabierają czas.

Kiedy już ustalimy sobie warunki, w których mamy otwierać obwódkę, musimy cały czas sztywno się ich trzymać. Jakakolwiek ich zmiana (np. przełączenie z grafiki wielokolorowej na jednobarwną) zmusi nas do powtórnego cyklowania całej procedury.

Aby wejść w przerwanie, które umożliwi nam zabawę z ramkami bocznymi, musimy przestawiać wektor IRQ dwukrotnie. Najpierw — do procedurki, która li tylko przestawi przerwania do właściwej procedury likwidowania ramki, a potem natychmiast przerwanie to uruchomi rozkazem CLI. Później traktujemy to pierwsze przerwanie (nazwijmy je IRQ1) jak zwykłą procedurę inicjalizującą zakończoną wieczną pętlą. Ważne tu jest, by w tej wiecznej pętli był chociaż jeden rozkąz NOP. Bez niego program nie działa. Jeżeli tak przygotujemy przerwania, komputer skacze do naszej procedurki z dokładnością do dwóch cykli. Tzn. skacze albo we właściwym miejscu, albo 1 cykl za wcześnie. Aby temu zaradzić, poświęcamy całą pierwszą linię. Kiedy za pomocą odpowiednich opóźnień dojdziemy do jej końca, wydajemy cykl rozkazów:

```
    LDA $d012
    CMP #$33
    BEQ aa
aa  ...
    ...
```

Gdzie tu sens, gdzie logika? Zapytać można, bo w końcu wiadomo, że jesteśmy właśnie na końcu linii $33, 51. Otóż nie zawsze wiadomo. Jeżeli bowiem w przerwanie wskoczyliśmy ten jeden cykl zbyt wcześnie, rozkaz BEQ zostanie wykonany i zajmie 2 cykle zegara. Jeżeli zaś wskoczyliśmy w odpowiednim cyklu, znajdziemy się już w linii $34, 52 a rozkaz BEQ wykonany nie zostanie i zajmie tylko 1 cykl. Ważne jest, żeby nie przekroczyć strony pamięci pomiędzy BEQ aa i aa!

Dalsze linie robimy już sztampowo. Montujemy za pomocą rejestru Y siedmiokrotną pętlę opóźniającą (podobnie, jak w kolorowych paskach!). Po niej wykonujemy zwężenie i bezpośrednio potem rozszerzenie ekranu —

```
LDA #$c0
STA $d016
LDA #$c8
STA $d016
```

Pomiędzy pętlą opóźniającą a powyższą sekwencją trzeba z reguły wstawić kilka cykli opóźnienia. Aby znaleźć prawidłową liczbę cykli radzę podczas prób nie zmieniać $d016, a kolor tła — $d021:

```
LDA #$00
STA $d021
LDA #$06
STA $d021
```

Następnie pomiędzy to, a naszą pętlę wstawiamy opóźnienia. Opóźniamy z reguły rozkazam NOP (2 cykle), BIT $00 (3 cykle), LDA #$00 (2 cykle) i LDA $00 (3 cykle) (te rozkazy to mój prywatny zestaw, proszę nie traktować go dogmatycznie). Regułą jest, że 1 cykl — to 1 kratka. Do pomocy służy zaprogramowana przeze mnie miarka, dzięki której łatwiej zorientować się, ile jeszcze zostało do końca linii. Trzeba doprowadzić do stanu, w którym czarny pasek będzie się zaczynał nad ostatnią z dziewiątek. Tak robimy po kolei, w każdej linijce.

Zdarzyć się może, że w trakcie prac nad kolejnymi linijkami stracimy synchronizację linii pierwszej. Zaradzić temu można przez dodanie kilku cykli na końcu naszej procedury. Dlaczego tak się dzieje — nie wiem.

Aha, jeszcze jedno! Pamiętacie zapewne, że co ósma linia jest krótsza od pozostałych. Podczas zabawy z ramką bywa to szczególnie dokuczliwe. Kiedy damy 0 cykli opóźnienia, zaczynamy jeszcze przed rozpoczęciem linii. Gdy damy NOP (dwa cykle) zaczynamy akurat po rozpoczęciu wyświetlania ramki. Wszystkie te niedogodności biorą się z konieczności odświeżania pamięci DRAM, co zajmuje czas procesora. Jedyną możliwością pośrednią jest więc 1 cykl. Możemy go uzyskać jedynie poprzez indeksowanie rejestrem X lub Y rozkazu STA $d016, służącego do rozszerzenia ekranu w poprzedniej linii. STA indeksowane ma właśnie 1 cykl więcej, niż STA bezwzględne.

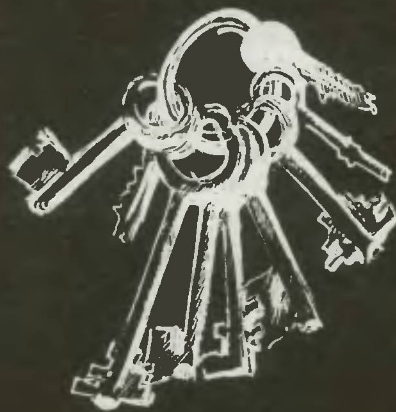
W pierwszej linii po linii krótszej cyklowanie jest jeszcze odrobinę inne niż w pozostałych. Trzeba o tym też pamiętać. Reguła jest więc taka — 1 krótka, 1 dłuższa i 6 normalnych, 1 krótka...

Przy wyjściu z przerwania trzeba pamiętać, że wychodzimy z przerwania wewnętrznego, czyli wywołanego innym przerwaniem. Przy takim układzie bardzo szybko przeładuje się stos i program nie będzie działał. Trzeba więc pamiętać o zdjęciu ze stosu sześciu liczb — trzech rejestrów (A, X, Y), rejestru słowa stanu procesora i licznika programu (2 liczby).

Jeśli szczęśliwie przebrnąłeś przez moje uwagi dotyczące likwidowania obwódki, musisz jeszcze tylko napisać odpowiedni program. Dla wprawki podałem przykład — program umożliwiający wpisanie na ramkę czterech jednokolorowych sprajtów (4, 5, 6, 7) na jednokolorowym tle. Jeżeli chciałbyś skorzystać z np. trybu wielokolorowego, musisz cały program pisać od nowa. Mam nadzieję, że zamieszczony listing przynajmniej pomoże Ci zrozumieć zasadę. Jeśli jednak jesteś zbyt zapracowany, by wpisać aż tak długi listing, zamów po prostu dyskietkę „C&A". Listingi zamieszczam na stronie 32.

I jeszcze jedna kwestia — płynące napisy na ramkach. Przepraszam, że nie zamieszczam listingu, ale drugi tak sążnisty program wystawiałby na próbę spokój ducha innych Czytelników. Postaram się jednak wyjaśnić główne założenia programu. Chodzi głównie o to, aby na wysokości napisu umieścić na ramce sprajty, na sprajtach zaś trzeba wstawić litery, które potem będziemy przesuwać. Ogólnie metody są dwie: prostsza polega na zwyczajnej manipulacji współrzędną poziomą duszka, natomiast skuteczniejsza — na przesuwaniu w lewo zawartości duszków za pomocą rozkazu ROL.

BARTEK KACHNIARZ



ZOSTAŃ „WŁAMYWACZEM"!

„...eee, coś Ty! Czytam te artykuły o asemblerze w C&A i niby rozumiem, ale tak właściwie to nie widzę sensu w uczeniu się tego wszystkiego. Pograć to lubię, ale nie mam zamiaru być programistą! A napis na ekranie wyświetlę z BASICA, bez tej chińszczyzny. Można go nawet od razu umieścić w dowolnym miejscu ekranu, bez używania tego Q! Wojtek pokazał mi, jak to zrobić, ale sam rozumiesz — on „siedzi" w BASICU już od pół roku."

„Eee, ale może udało by się znaleźć nieśmiertelność do COMMANDO, bo dostanę szału jak tego nie skończę."

„Eee, a wiesz, jak się do tego zabrać?"

„No, eee, nie bardzo... ale inni jakoś to robią."

Stałem na krakowskiej giełdzie oczekując na kolegę i przypadkowo usłyszałem fragment rozmowy świeżo upieczonych właścicieli C-64. Bliżej nieokreślone znaczeniowo wyrażenie „eee" świetnie oddawało, zależnie od intonacji, stan duszy rozmówców. Pomyślałem sobie, że jest to klasyczny przykład średniej statystycznej. No bo nie oszukujemy się! Większości posiadaczy komputery służą niemal wyłącznie do grania! Czasami właściciel, który liznął nieco BASIC, wklepie jakiś ciekawy program i właściwie to koniec jego wiedzy. A ci nieliczni, którzy chcieliby trochę się podszkolić, poznać asembler i zrobić z tej znajomości użytek, nie wiedzą, jak powiązać przyjemne z pożytecznym, gdyż jawi im się on jako abrakadabra dostępna tylko wybrańcom.

Tymczasem problem polega nie tyle na pamięciowym opanowaniu asemblera, ile na braku umiejętności powiązania teorii z praktyką. Jak w kilkudziesięciu kB programu gry odszukać „życie", „energię" i inne, ułatwiające ukończenie gry informacje? Spróbuję nieco naświetlić te zagadnienia i wskazać kilka sposobów postępowania, które zazwyczaj kończą się sukcesem. Ten rodzaj pracy (tak, tak — jest to praca, czasem dość trudna) z komputerem doskonale ułatwia opanowanie asemblera, którego stopniowo rosnąca znajomość pozwala na coraz łatwiejsze poruszanie się w dżungli rozpościerającej się od adresu $0000 do $FFFF. Otóż to: asemblera można nauczyć się przez dokonywanie „włamań" do programów gier, dzięki efektowi dodatniego sprzężenia zwrotnego! Tymczasem większość z Was sądzi, że kolejność musi być odwrotna — najpierw trzeba znać asembler, aby móc myśleć o wprowadzaniu zmian do programów gier.

Proszę mi wierzyć, taki sposób nauki asemblera jest nie tylko możliwy, ale i najłatwiejszy. Dodatkową zaletą jest połączenie nauki z przyjemnością oraz konkretnymi sukcesami w grach, o których ukończeniu trudno byłoby nawet marzyć. Niedowiarkom oświadczam, że zaczynałem ze znajomością mnemoników LDA i STA oraz ich trybów adresowania. Skończyło się na samodzielnym rozpracowaniu kilkudziesięciu gier, w tym kilkunastu całodyskowych. I nie oznacza to wcale, że „włamaniom" poświęcałem cały wolny czas.

Zakładam, że znasz jako tako BASIC, potrafisz przeliczyć adres szesnastkowy na dziesiętny i odwrotnie oraz umiesz zrobić użytek z monitora lub asemblera, w które wyposażone są moduły typu FINAL czy ACTION REPLAY. Byłoby też wskazane, abyś chociaż pobieżnie coś poczytał o asemblerze 6502.

Zanim zajmiemy się „życiami", „energią" itp., trzeba nauczyć się zatrzymywania wykonywanego programu gry celem umożliwienia jego odczytu (zdekodowania) oraz ponownego uruchomienia ale już PO wprowadzeniu poprawek, które ułatwią lub wręcz umożliwią jej ukończenie.

Przyjmij na wiarę, że zatrzymanie wykonywania programu maszynowego realizowane jest na poziomie sprzętowym. W tym celu wykorzystywane są przerwania wraz z odpowiednimi procedurami ich obsługi. Bardzo cenną cechą C-64, przynajmniej z punktu widzenia włamywacza, jest jego zdolność do utrzymania praktycznie nie zmienionej zawartości pamięci RAM po zatrzymaniu wykonywania programu.

Wspomniane już moduły mają wbudowane przyciski RESET i FREEZE, służące do zatrzymywania programu. Pojawia się pytanie: czym różnią się efekty ich użycia, skoro służą temu samemu celowi? Nie wdając się w szczegóły można powiedzieć, że wciśnięcie RESET powoduje wykonanie całego szeregu procedur inicjalizacyjnych, zawartych w ROM, tak jak to ma miejsce bezpośrednio po włączeniu komputera. Zawartość RAM pozostaje prawie nie zmieniona. Prawie, gdyż procedury te wpisują do niektórych komórek RAM bajty o charakterystycznych wartościach. Wciśnięcie FREEZE powoduje pojawienie się i obsługę przerwania niemaskowalnego NMI, co w efekcie również umożliwia dostęp do zawartości RAM, jednak ilość komórek pamięci, które zmieniły wartość, jest znacznie mniejsza. Czasami może to mieć decydujące znaczenie dla ponownego uruchomienia gry. W najlepszym położeniu będą posiadacze ACTION REPLAY MK V, którzy mogą dokonać takiego uruchomienia nawet w wypadku gier dyskietkowych. Umożliwia to jego rozwiązanie sprzętowe. Do wbudowanej w module niewielkiej ilości RAM zapisywana jest zawartość wszystkich rejestrów procesora, licznik programu, wektory przerwań. Dysponując tymi danymi z łatwością uruchomisz każdy program. Jest to bardzo wygodne, lecz pozbawia Cię jednego z ważnych etapów oswojenia się z asemblerem i obsługą modułu. Szczerze radzę nie iść na łatwiznę, przynajmniej z początku, gdyż zdobyte doświadczenie i rutyna będą procentować później. Gdy

już wystarczająco poznasz asembler i będziesz chciał pracować nad grami dyskietkowymi, ACTION REPLAY zaoszczędzi Ci mnóstwo czasu. Przyjmijmy więc zasadę: gry dyskietkowe uruchamiamy za pomocą ACTION REPLAY, natomiast gry taśmowe — „ręcznie". Aby tego dokonać, musimy nauczyć się odnajdywania adresu początkowego.

ADRES POCZĄTKOWY

Załóżmy, że mamy wczytany do pamięci komputera program (mowa tu o programach zapisanych w turbo lecz bez procedur samouruchamiających). Na ekranie widać napisy:

FOUND tytuł programu
LOADING
READY.

Zamiast F3 (RUN) wciskamy F1 (LIST) i najczęściej naszym oczom ukaże się nn SYS mmmm, gdzie nn — numer wiersza (dziesiętnie) w BASIC, a mmmm — adres (dziesiętnie) procedury uruchamiającej. Zapisujemy ten adres i wciskamy F3. Jeżeli gra uruchomi się natychmiast, to jest duża szansa, że odnaleziony tak adres początkowy jest tym, czego szukamy. Najczęściej jednak na ekranie ukaże się plątanina znaków, którym towarzyszą efekty dźwiękowe. Po kilku naciśnięciach spacji i zmianach wyświetlanego obrazu gra uruchamia się. Co się stało?

Takie efekty świadczą o tym, że program gry poddany był obróbce programem narzędziowym, tzw. cruncherem. Efektem działania takiego programu jest zmniejszenie objętości programu głównego, który zajmuje dzięki temu mniej miejsca na taśmie. Tak skompresowany program należy, po wprowadzeniu do pamięci komputera, „rozpakować" i umieścić jednocześnie we właściwym obszarze pamięci RAM. Jest to bardzo ważne, gdyż C-64 umożliwia pracę — zależnie od konkretnych potrzeb programisty — w jednym z czterech banków pamięci po 16 kB każdy (tyle jest bowiem w stanie zaadresować procesor graficzny VIC. Wszystkich tych operacji dokonuje procedura w asemblerze, której adres startowy znaleźliśmy przyciskiem F1. Niestety, z reguły adres ten nie pokrywa się z adresem początkowym zdekompresowanego już programu. I co teraz?

Zacznijmy od zdefiniowania czego mamy szukać. Adres początkowy jest adresem komórki pamięci, w której znajduje się początek programu. Innymi słowy, jeśli zaczniemy wykonywać program zaczynając od tej właśnie komórki, to uruchomimy grę. Z poziomu BASIC uzyskuje się to poprzez wpisanie: SYS adres i wciśnięcie RETURN.

Jak więc odszukać adres początkowy już zdekompresowanego i uruchomionego programu? Zacznę może od niezbyt pocieszającego i odkrywczego stwierdzenia, że nie ma na to uniwersalnego sposobu. Można jednak przyjąć pewien schemat postępowania. Niemałą rolę odgrywa tu praktyka i wynikająca z niej znajomość manier i sztuczek programistów. Mogę śmiało powiedzieć, że co najmniej 80% programów rozpoczyna się rozkazem SEI — wyłącz przerwania. Reprezentuje go bajt o wartości $78 (120 dziesiętnie). SEI należy do grupy rozkazów jednobajtowych.

Tak więc po uruchomieniu programu naciskamy FREEZE i przechodzimy do monitora. Aby uzyskać dostęp do całej pamięci RAM wpisz 04 i wciśnij RETURN. Zwracam uwagę, że wpisujemy literę, a nie cyfrę! Przeszukaj pamięć starając się znaleźć rozkaz SEI przez wpisanie H 0000 FFFF 78 (i RETURN). Po chwili monitor wyświetli w postaci szesnastkowej adresy komórek pamięci, w których umieszczona jest wartość $78. Nie oznacza to wcale, że w programie jest tak dużo rozkazów SEI. Należy teraz sprawdzić, czy po zdekodowaniu programu w dół i w górę od znalezionego adresu (musi on „wyjść" poza ekran) w programie nadal jako samodzielny rozkaz figuruje SEI, czy też został on „wchłonięty" przez poprzedzający lub następny rozkaz asemblera. Wykonujemy to przez wpisanie D adres i używając F3 i F5 dekodujemy program w górę i w dół. Najczęściej 3–4 uprzednio znalezione adresy dają w efekcie „sensowny" wygląd programu zaczynającego się rozkazem SEI. który z nich jest właściwy? To można sprawdzić tylko metodą prób i błędów. Przejdź do BASIC wpisując X. Po READY wpisujemy SYS $adres i wciśnij RETURN. Sukces! W przypadku niepowodzenia należy sprawdzić pozostałe adresy, za każdym razem (niestety!) wczytując grę od nowa. Przeważnie któraś z prób kończy się sukcesem.

Gdy podczas prób ukazał się chociaż fragment obrazu pojawiającego się na ekranie przy normalnym uruchamianiu gry, to jesteśmy blisko celu. Należy teraz (po kolejnym wczytaniu) zdekodować program od adresu, który dał efekt częściowego uruchomienia, np. od $1234 i sprawdzić, czy kilkadziesiąt (kilkaset) bajtów dalej napotkamy rozkaz RTS (powrót z podprogramu). Teraz poszukaj w pamięci podprogramu zaczynającego się od tego adresu, czyli JSR $1234. Jeżeli monitor wyświetlił jeden adres, to niemal na pewno jest on poszukiwanym adresem startowym, a program może wyglądać np. tak:

```
1234 SEI
1235 LDA #$35
1237 STA $01
.
.
.
1293 RTS  (tu program przy starcie od $1234
      „powiesił" się)
.
.
.
2345 JSR $1234 (adres startowy)
2348 LDA #$FF
234A STA $5681 itd.
```

Może się też okazać, że właściwy do poszukiwań rozkazu JSR adres znajduje się kilkanaście (kilkadziesiąt) bajtów przed rozkazem SEI, co należy cierpliwie sprawdzić.

Jeżeli poszukiwanie JSR nie dało rezultatu, to sprawdzamy wg analogicznego schematu, występowanie instrukcji skoku JMP. Należy jedynie pamiętać, że poszukiwany adres początkowy znajduje się powyżej adresu skoku JMP. Najlepiej wyjaśni to przykład:

```
1234 SEI
1235 LDA #$35
1237 STA $01
.
.
.
3455 RTS
3456 JSR $27CE (adres początkowy)
3459 LDA #$00
345B STA $B5
345D JMP $1234 itd.
```

Gdy oba powyższe warianty zawodzą, pozostaje nam przeszukanie stosu i znalezienie na nim rozkazów JMP. Często (na nasze szczęście) procedura uruchamiająca grę odkłada na stosie adres początkowy jako JMP $adres.

Stos przeszukujemy wpisując H 0100 0200 4 C i sprawdzamy, czy ewentualnie odnalezione w ten sposób adresy spełniają nasze nadzieje. Gdy opisane powyżej trzy metody zawiodą, pozostaje sprawdzenie zawartości komórek pamięci $0303 i $0304. Niektóre procedury uruchamiające umieszczają w nich wektor startu kolejności młodszy bajt, starszy bajt.

Ostatnią „naukową" metodą poszukiwania adresu startowego jest skorzystanie z informacji o zmianach standardowych wektorów przerwań IRQ i NMI przez program. Operacja ta, ze zrozumiałych względów, musi być dokonana gdzieś na początku programu. Przeszukujemy więc pamięć pod kątem rozkazu STA $FFFF, czyli sekwencji bajtów 8D FF FF i — jeżeli znajdziemy gdzieś w pamięci sekwencję rozkazów typu:

```
4567 LDA #$7C
4569 STA $FFFF
456D LDA #$08
456F STA $FFFE
```

to (poza informacją o aktualnym wektorze przerwań) istnieje szansa, że adres początkowy znajduje się powyżej niej.

Gdy wszystko zawiodło, pozostaje już tylko intuicja i rutyna. Ale i im można pomóc. Z własnych doświadczeń wiem, że należy sprawdzić całość programu pod kątem występowania w nim dłuższej sekwencji rozkazów JMP lub JSR. Niektórzy programiści wręcz uwielbiają start z takiej „tablicy" skoków lub podprogramów. Czasem z pierwszego, czasem z ostatniego lub też ze środkowego. Warto też rutynowo wykorzystywać metodę sprzętową. Czasami za podawanym jawnie adresem procedury dekompresującej (F1 — LIST) programista podaje również adres startowy programu już po dekompresji, ale jest on zabezpieczony przed wyświetleniem na ekranie. Użycie instrukcji OLD z reguły łamie te zabezpieczenia i po LIST wyświetlana jest niewidoczna wcześniej informacja. A więc RESET, F4 i F1. Przeciętnie raz na dziesięć ujrzymy to, czego moglibyśmy długo szukać.

Gry, w których ponowne uruchomienie po przejściu do monitora jest niemożliwe spotyka się bardzo rzadko. Sytuacja taka ma miejsce w przypadku gdy modyfikacja wektorów przerwań następuje nie przez program, ale jest dokonywana bezpośrednio przez procedurę dekompresującą. Aby ponownie uruchomić taki program można spróbować dokonać tego przez dopisanie własnych procedur ustawiających właściwie wektory przerwań lub używając ACTION REPLAY.

cdn.

URAN

BASIC pod KLAWISZEM

Zapewne każdy z Was pamięta obsługę „staruszka Spectrum. Klawiaturze tego komputera przyporządkowane były całe instrukcje języka BASIC. Spróbujmy zrealizować coś takiego dla C-64.

Poniżej przedstawiam program (listing 1), po uruchomieniu którego, po naciśnięciu kombinacji klawiszy COMMODORE LOGO KEY i CONTROL, kursor zmieni kolor. Teraz, jeżeli naciśniemy np. klawisz „I", na ekranie pojawi się cała instrukcja „INPUT". A oto wyszczególnienie, jakim klawiszom przyporządkowane są poszczególne instrukcje.

KLAWISZ		INSTRUKCJA
A	—	„AND"
B	—	„CONT"
C	—	„CHR$("
D	—	„DATA"
E	—	„END"
F	—	„FOR"
G	—	„GOTO"
H	—	„GET"
I	—	„INPUT"
J	—	„LOAD"
K	—	„REM"
L	—	„LIST" + RETURN
M	—	„READ"
N	—	„NEXT"
O	—	„OPEN"
P	—	„POKE"
Q	—	„PEEK"
R	—	„RETURN"
S	—	„SAVE"@:"
T	—	„THEN"
U	—	„USR"
V	—	„VERIFY"
W	—	„WAIT"
X	—	„SYS"
Y	—	„RND"
Z	—	„RUN" + RETURN

Aby użytkownik mógł łatwo przystosować program do własnych potrzeb, przedstawiam także listing źródłowy programu (listing 2) w asemblerze (TURBO ASSEMBLER 5). Należy także pamiętać, iż instrukcja przyporządkowana danemu klawiszowi nie może przekraczać dziewięciu znaków. W praktyce, modyfikując program możemy mieć nieograniczoną liczbę klawiszy funkcyjnych. Program nie należy do najkrótszych, ale za to można go w bardzo prosty sposób modyfikować.

MARIUSZ FERDYN

OD REDAKCJI:
Ze względu na długość listingu źródłowego zdecydowaliśmy, że będzie on dostępny jedynie na dyskietce „C&A". Za tę niedogodność serdecznie Czytelników przepraszamy.

```
200 rem ***************************
205 rem *            keys            *
215 rem *          m.ferdyn          *
220 rem ***************************
225 d=49920:b=72319
230 c=0:e=d
235 read a$:if a$="end" then 285
240 a1=asc (left$ (a$,1)) and 63
245 a2=asc (right$ (a$,1)) and 63
250 if a1>47 then 260
255 a1=a1+9:goto 265
260 a1=a1-48
265 if a2>47 then a2=a2-48:goto 2
    75
270 a2=a2+9
275 a=a1*16+a2:poke d,a
280 d=d+1:c=c+a:goto 235
285 if c<>b then print "blad w li
    niach data":stop
290 sys e
295 print chr$ (147)
300 print:print "use commodore lo
    go key + control"
305 print "                     and
    keys a,b,c...z.":new
306 :
310 data 78,ad,14,03,8d,34,03,ad
315 data 15,03,8d,35,03,a9,1e,8d
320 data 14,03,a9,c3,8d,15,03,a9
325 data 00,8d,5d,c5,58,60,ad,5d
330 data c5,c9,01,f0,15,ad,8d,02
335 data c9,06,f0,03,6c,34,03,a9
340 data 01,8d,5d,c5,ee,86,02,6c
345 data 34,03,a5,cb,c9,40,f0,f7
350 data c9,0a,f0,67,c9,1c,f0,66
355 data c9,14,f0,65,c9,12,f0,64
360 data c9,0e,f0,63,c9,15,f0,62
365 data c9,1a,f0,61,c9,1d,f0,60
370 data c9,21,f0,5f,c9,22,f0,5e
375 data c9,25,f0,5d,c9,2a,f0,5c
380 data c9,24,f0,5b,c9,27,f0,5a
385 data c9,26,f0,59,c9,29,f0,58
390 data c9,3e,f0,57,c9,11,f0,56
395 data c9,0d,f0,55,c9,16,f0,54
400 data c9,1e,f0,53,c9,1f,f0,52
405 data c9,09,f0,51,c9,17,f0,50
410 data c9,19,f0,4f,c9,0c,f0,4e
415 data 4c,f9,c3,4c,04,c4,4c,0b
420 data c4,4c,12,c4,4c,19,c4,4c
425 data 20,c4,4c,27,c4,4c,2e,c4
430 data 4c,35,c4,4c,3c,c4,4c,43
435 data c4,4c,4a,c4,4c,51,c4,4c
440 data 58,c4,4c,5f,c4,4c,66,c4
445 data 4c,6d,c4,4c,74,c4,4c,7b
450 data c4,4c,82,c4,4c,89,c4,4c
455 data 90,c4,4c,97,c4,4c,9e,c4
460 data 4c,a5,c4,4c,ac,c4,4c,b3
465 data c4,a9,00,8d,5d,c5,ce,86
470 data 02,6c,34,03,a2,d8,a0,c4
475 data 4c,b7,c4,a2,dc,a0,c4,4c
480 data b7,c4,a2,e1,a0,c4,4c,b7
485 data c4,a2,e7,a0,c4,4c,b7,c4
490 data a2,ec,a0,c4,4c,b7,c4,a2
495 data f0,a0,c4,4c,b7,c4,a2,f4
500 data a0,c4,4c,b7,c4,a2,f9,a0
505 data c4,4c,b7,c4,a2,fd,a0,c4
510 data 4c,b7,c4,a2,03,a0,c5,4c
515 data b7,c4,a2,08,a0,c5,4c,b7
520 data c4,a2,0c,a0,c5,4c,b7,c4
525 data a2,12,a0,c5,4c,b7,c4,a2
530 data 17,a0,c5,4c,b7,c4,a2,1c
535 data a0,c5,4c,b7,c4,a2,21,a0
540 data c5,4c,b7,c4,a2,26,a0,c5
545 data 4c,b7,c4,a2,2b,a0,c5,4c
550 data b7,c4,a2,32,a0,c5,4c,b7
555 data c4,a2,3a,a0,c5,4c,b7,c4
560 data a2,3f,a0,c5,4c,b7,c4,a2
565 data 43,a0,c5,4c,b7,c4,a2,4b
570 data a0,c5,4c,b7,c4,a2,50,a0
575 data c5,4c,b7,c4,a2,54,a0,c5
580 data 4c,b7,c4,a2,58,a0,c5,8e
585 data c5,c4,8c,c6,c4,a9,14,8d
590 data 77,02,a2,00,bd,00,10,c9
595 data 00,f0,07,9d,78,02,e8,4c
600 data c4,c4,e8,86,c6,4c,f9,c3
605 data 41,4e,44,00,43,4f,4e,54
610 data 00,43,48,52,24,28,00,44
615 data 41,54,41,00,45,4e,44,00
620 data 46,4f,52,00,47,4f,54,4f
625 data 00,47,45,54,00,49,4e,50
630 data 55,54,00,4c,4f,41,44,00
635 data 52,45,4d,00,4c,49,53,54
640 data 0d,00,52,45,41,44,00,4e
645 data 45,58,54,00,4f,50,45,4e
650 data 00,50,4f,4b,45,00,50,45
655 data 45,4b,00,52,45,54,55,52
660 data 4e,00,53,41,56,45,22,40
665 data 3a,00,54,48,45,4e,00,55
670 data 53,52,00,56,45,52,49,46
675 data 59,22,00,57,41,49,54,00
680 data 53,59,53,00,52,4e,44,00
685 data 52,55,4e,0d,00,00,end
```

KOLORKI

Oglądając grafikę jaką generuje Amiga, z pewnością zauważyłeś ogromną ilość kolorów. Jest czego zazdrościć, ponieważ 4096 to naprawdę niemało. Przy tym Commodore 64 ze swoimi szesnastoma kolorami to tak jak trabant przy ferrari. Nie załamujmy jednak rąk, gdyż można coś na to poradzić.

Na pewno bawiłeś się kiedyś farbami. Mieszając farby uzyskiwało się kolory pośrednie. Proste, ale co to ma wspólnego z C-64? Otóż spróbujemy zwiększyć ilość kolorów przez mieszanie ich. Jak to uzyskać? Nic prostszego! Jeżeli będziemy bardzo szybko na przemian wyświetlać dwa kolory, to uzyskamy kolor pośredni. Rozwiązanie takie pozwala uzyskać 136 kolorów, lecz nie jest ono pozbawione wad. Otóż wyświetlany w ten sposób obraz drży i bardzo szybko męczy oczy. Najgorzej gdy mieszamy kolory, których jasności znacznie się różnią, np. czarny i biały, natomiast przy mieszaniu kolorów o niewielkiej różnicy jasności migotanie jest prawie niezauważalne.

Jeżeli będziemy brali pod uwagę ten fakt tworząc grafikę, to możemy uzyskać bardzo ciekawe rezultaty. Niestety nie spotkałem, jak na razie, programu, który umożliwiałby tworzenie grafiki z wykorzystaniem tej metody.

Poniższy program pokazuje przykładowe przejścia kolorów uzyskane tą metodą. Inne przykłady, znacznie lepiej odzwierciedlające możliwości takiego rozwiązania znaleźć można w demku grupy Origo pt. ##############.

RAFAŁ PIASEK

```
200 rem ****************************
205 rem * coded by jetboy/cavern *
210 rem ****************************
215 d=49152:b= 54096
220 c=0:e=d
225 read a$:if a$="end" then 275
230 a1=asc(left$(a$,1))and63
235 a2=asc(right$(a$,1))and63
240 if a1>47 then 250
245 a1=a1+9:goto 255
250 a1=a1-48
255 if a2>47 then a2=a2-48:goto265
260 a2=a2+9
265 a=a1*16+a2:poked,a
270 d=d+1:c=c+a:goto225
275 if c<>b then print"blad w lini
    ach data":stop
280 sys e
285 data 78,20,5b,ff,a2,00,8e,20
290 data d0,8e,21,d0,a2,c0,a0,40
295 data 8c,14,03,8e,15,03,a2,1b
300 data 8e,11,d0,a2,f1,8e,1a,d0
305 data a2,7f,8e,0d,dc,a2,00,8e
310 data 0e,dc,a2,fa,8e,12,d0,58
315 data a2,00,bd,70,c0,9d,00,04
320 data e8,e0,f0,d0,f5,4c,3d,c0
325 data a9,01,8d,19,d0,ad,6f,c0
330 data 29,01,f0,10,a2,00,bd,60
335 data c1,9d,00,d8,e0,f0,e8,d0
340 data f5,4c,69,c0,a2,00,bd,50
345 data c2,9d,00,d8,e8,e0,f0,d0
350 data f5,ee,6f,c0,4c,31,ea,00
355 data 0b,09,0c,0b,01,20,0b,0f
360 data 0c,0f,12,0f,17,20,17,09
365 data 05,03,05,0a,20,20,20,03
370 data 26,01,20,31,39,39,32,20
375 data 20,20,70,72,40,72,40,6e
380 data 40,40,40,40,40,40,40,40
385 data 40,40,40,40,40,40,40,40
390 data 40,40,40,40,20,20,20,20
395 data 20,20,20,20,20,20,20,20
400 data 20,20,6b,71,72,71,72,73
405 data f7,f7,f7,f7,f7,f7,f7,f7
410 data f7,f7,f7,f7,f7,f7,f7,f7
415 data f7,f7,f7,f7,f7,f7,f7,f7
420 data f7,f7,f7,f7,f7,f7,f7,f7
425 data f7,f7,6b,72,71,72,71,73
430 data f7,f7,f7,f7,f7,f7,f7,f7
435 data f7,f7,f7,f7,f7,f7,f7,f7
440 data f7,f7,f7,f7,f7,f7,f7,f7
445 data f7,f7,f7,f7,f7,f7,f7,f7
450 data f7,f7,6b,71,72,71,72,73
455 data f7,f7,f7,f7,f7,f7,f7,f7
460 data f7,f7,f7,f7,f7,f7,f7,f7
465 data f7,f7,f7,f7,f7,f7,f7,f7
470 data f7,f7,f7,f7,f7,f7,f7,f7
475 data f7,f7,6b,72,71,72,71,73
480 data 04,0f,02,12,1a,05,20,20
485 data 20,20,20,20,20,20,20,20
490 data 20,20,20,20,20,20,20,20
495 data 20,20,20,1a,0c,05,21,21
500 data 21,20,6d,71,40,71,40,7d
505 data 06,06,04,04,0e,0e,05,05
510 data 03,03,0d,0d,01,01,01,0d
515 data 0d,03,03,05,05,0e,0e,04
520 data 04,06,06,04,04,0e,0e,05
525 data 05,20,04,04,02,02,06,06
530 data 09,09,02,02,08,08,0a,0a
535 data 0f,0f,07,07,01,01,01,07
540 data 07,0f,0f,0a,0a,08,08,02
545 data 02,09,09,09,02,02,08,08
550 data 0a,20,0a,04,04,02,02,06
555 data 06,02,02,04,04,0e,0e,05
560 data 05,03,03,0d,0d,01,01,01
565 data 0d,0d,03,03,05,05,0e,0e
570 data 04,04,02,02,06,06,06,02
575 data 02,20,0a,0a,04,04,02,02
580 data 09,09,08,08,0c,0c,0f,0f
585 data 0d,0d,01,01,01,0d,0d,0f
590 data 0f,0c,0c,08,08,09,09,09
595 data 08,08,0c,0c,0f,0f,0d,0d
600 data 01,20,0f,0a,0a,04,04,02
605 data 20,20,20,20,20,20,20,20
610 data 20,20,20,20,20,20,20,20
615 data 20,20,20,20,20,20,20,20
620 data 20,20,20,20,20,20,20,20
625 data 20,20,0f,0f,0a,0a,04,04
630 data 05,05,05,05,05,05,20,20
635 data 20,20,20,20,20,20,20,20
640 data 20,20,20,20,20,20,20,20
645 data 20,20,20,01,01,01,01,01
650 data 01,20,0d,0f,0f,0a,0a,04
655 data 20,06,06,04,04,0e,0e,05
660 data 05,03,03,0d,0d,01,01,01
665 data 0d,0d,03,03,05,05,0e,0e
670 data 04,04,06,06,04,04,0e,0e
675 data 05,20,0a,04,04,02,02,06
680 data 09,02,02,08,08,0a,0a,0f
685 data 0f,07,07,01,01,01,07,07
690 data 0f,0f,0a,0a,08,08,02,02
695 data 09,09,09,02,02,08,08,0a
700 data 0a,20,0a,0a,04,04,02,02
705 data 06,06,02,02,04,04,0e,0e
710 data 05,05,03,03,0d,0d,01,01
715 data 01,0d,0d,03,03,05,05,0e
720 data 0e,04,04,02,02,06,06,06
725 data 02,20,0f,0a,0a,04,04,02
730 data 09,08,08,0c,0c,0f,0f,0d
735 data 0d,01,01,01,0d,0d,0f,0f
740 data 0c,0c,08,08,09,09,09,08
745 data 08,0c,0c,0f,0f,0d,0d,01
750 data 01,20,0f,0f,0a,0a,04,04
755 data 20,20,20,20,20,20,20,20
760 data 20,20,20,20,20,20,20,20
765 data 20,20,20,20,20,20,20,20
770 data 20,20,20,20,20,20,20,20
775 data 20,20,0d,0f,0f,0a,0a,04
780 data 01,0d,03,05,0e,04,20,20
785 data 20,20,20,20,20,20,20,20
790 data 20,20,20,20,20,20,20,20
795 data 20,20,20,20,20,20,20,20
800 data 20,20,0d,0d,0f,0f,0a,0a
805 data end
```

FRAKTALE

```
100 rem *******************
105 rem *                 *
110 rem *    fraktale     *
115 rem *                 *
120 rem *   przemyslaw    *
125 rem *  kozankiewicz   *
130 rem *                 *
135 rem *******************
140 :
145 poke 53280,0:poke 53281,0:pok
    e 646,7
150 :
155 rem wczytanie danych
160 print chr$(147)"ilosc podobie
    nstw";:input ns
165 dim si(2,3,ns),pr(ns)
170 input"ilosc punktow";it
175 gosub 360:rem odczyt przekszt
    alcen
180 :
185 rem obliczanie prawdopodobien
    stws wylosowania przeksztalce
    nia
190 for i=1 to ns:pr(i)=abs(si(1,
    1,i)*si(2,2,i)-si(2,1,i)*si(1
    ,2,i))+.03
195 pr=pr+pr(i)
200 next i
205 p=0
210 for i=1 to ns:p=p+pr(i):pr(i)
    =p/pr:next
215 :
220 rem rysowanie fraktals
225 :
230 hires 1,0:rem graphic on
235 x=0:y=0
240 for ii=1 to it
245 :
250 rem losowanie jednego z przek
    sztalcen
255 p=rnd(1)
260 for i=1 to ns:k=i:if p<pro(i)
    then i=ns
265 next
270 :
275 rem przeksztalcenie punktu
280 xn=x*si(1,1,k)+y*si(1,2,k)+si
    (1,3,k)
285 yn=x*si(2,1,k)+y*si(2,2,k)+si
    (2,3,k)
290 x=xn:y=yn:if ii>20 then gosub
    335
295 next
300 :
305 poke 53280,2
310 get a$:if a$="" then 310
315 end
320 :
325 rem procedury pomocnicze
330 :
335 rem wstaw punkt
340 if x<0 or x>320 or y<0 or y>2
    00 then return
345 plot x,y,1
350 return
355 :
360 rem wczytanie przeksztalcen i
    obliczenie stalych
365 for i=1 to ns
370 nrm:print chr$(147)"przekszta
    lcenie#"i
375 input "0,0      ->";x1,y1
380 input "320,0    ->";x2,y2
385 input "0,200    ->";x3,y3
390 si(1,1,i)=(x2-x1)/320
395 si(1,2,i)=(x3-x1)/200
400 si(1,3,i)=x1
405 si(2,1,i)=(y2-y1)/320
410 si(2,2,i)=(y3-y1)/200
415 si(2,3,i)=y1
420 :
425 rem wykresl przeksztalcenie e
    kranu
430 restore
435 for a=0 to 3:read x,y
440 xn=x*si(1,1,i)+y*si(1,2,i)+si
    (1,3,i)
445 yn=x*si(2,1,i)+y*si(2,2,i)+si
    (2,3,i)
450 print xn,yn
455 if a=0 then x0=xn:y0=yn
460 if a=1 thenx1=xn:y1=yn
465 if a=2 thenx2=xn:y2=yn
470 if a=3 thenx3=xn:y3=yn
475 next
480 :
485 hires 1,0
490 xp=x0:yp=y0:xk=x1:yk=y1:gosub
    545
495 xp=x2:yp=y2:xk=x1:yk=y1:gosub
    545
500 xp=x2:yp=y2:xk=x3:yk=y3:gosub
    545
505 xp=x0:yp=y0:xk=x3:yk=y3:gosub
    545
510 get a$:if a$="" then 510
515 if a$="n" then 370
520 nrm:next
525 return
530 :
535 rem dane wspolrzednych ekranu
540 data 0,0,320,0,320,200,0,200
545 :
550 rem kreslenie linii
555 rem o koncach rzeczywistych
560 if xp>xk then a=xp:xp=xk:xk=a
    :a=yp:yp=yk:yk=a
565 dx=abs(xk-xp):dy=abs(yk-yp)
570 if dy>dxthen610
575 if xp>xk then a=xp:xp=xk:xk=a
    :a=yp:yp=yk:yk=a
580 a=(yk-yp)/(xk-xp)
585 for x=xp to xk step 8
590 y=yp+a*(x-xp)
595 gosub 340
600 next
605 return
610 if yp>yk then a=xp:xp=xk:xk=a
    :a=yp:yp=yk:yk=a
615 a=(xk-xp)/(yk-yp)
620 for y=yp to yk step 5
625 x=xp+a*(y-yp)
630 gosub 340
635 next
640 return
```

„Fraktale" to termin znany już od dawna, jednak swą popularność zyskał dopiero przed kilkunastu laty. Główną cechą fraktali jest ich „samopodobieństwo" (w każdym fraktalu można odnaleźć fragmenty podobne do jego całości lub jakiejś większej części).

Program FRAKTALE został napisany w języku SIMON'S BASIC (ze względu na grafikę wysokiej rozdzielczości). Dostosowanie go do innych programów umożliwiających użycie tego trybu w BASIC nie powinno sprawić problemu. FRAKTALE korzystają tylko z dwóch instrukcji graficznych;

HIRES col1, col2 — inicjacja trybu graficznego,
PLOT x,y,1 — włączenie punktu o współrzędnych (x,y).

Algorytm FRAKTALI wykorzystuje wiadomości z teorii przekształceń płaszczyznowych. W programie tym określają one podobieństwa wybranych części fraktala do jego całości.

Po uruchomieniu program pyta o liczbę przekształceń, które chcemy zdefiniować oraz o liczbę punktów, jaką ma wyświetlić (oczywiście, im więcej punktów, tym rysunek jest wyraźniejszy, lecz kreślenie trwa dłużej). Zadowalający wynik można zwykle uzyskać już przy 5 tysiącach punktów.

Następnie musimy określić kolejno wszystkie przekształcenia, tzn. jaka część rysunku na ekranie ma być podobna do jego całości. Program pyta o współrzędne punktów charakteryzujących przekształcenia. Punkty te odpowiadają trzem wierzchołkom ekranu: (320,0), (0,0), (0,200). Przykładem może tu być przekształcenie zwężające ekran: 0,0 → 10,0320,0 → 310.00,200 → 10,200

Po wprowadzeniu trzech współrzędnych punktów komputer kreśli obraz całego ekranu w tym przekształceniu. Wciśnięcie klawisza „N" daje możliwość określenia danego odwzorowania od nowa. Natomiast użycie innego klawisza powoduje przejście do następnego etapu.

Po wprowadzeniu wszystkich danych, zacznie się kreślenie naszego upragnionego fraktala.

A oto przykłady danych dla różnych fraktali:

1. ***„Liść paproci"***: 4 przekształcenia
   I: (80,15), (325,25), (70,175)
   II: (0,90), (80,87), (0,93)
   III: (15,85), (70,-10), (50,103)
   IV: (85,77), (150,177), (50,102)

2. ***„Trójkąt"***: 3 przekształcenia
   I: (0,100), (160,100), (0,200)
   II: (160,100), (320,100), (160,200)
   III: (80,0), (240,0), (80,100)

3. ***„Głowa mrówki"***: 3 przekształcenia
   I: (100,50), (220,50), (100,150)
   II: (0,100), (100,0), (100,200)
   III: (320,100), (220,0), (220,200)

FRAKTALE zostały napisane na podstawie artykułu „Jak pamiętać obrazy?" z miesięcznika DELTA 10/1989. Mają one własną procedurę kreślenia linii (polecenie LINE z SIMON'S BASIC nie przyjmuje współrzędnych leżących poza ekranem graficznym).

PRZEMYSŁAW KOZANKIEWICZ

RENUMBER C-64

Czasami istnieje potrzeba przenumerowania linii programu napisanego w BASIC, np. w celu połączenia dwóch programów. Z punktu widzenia użytkownika wystarczy mieć moduł nieco wyższej klasy (FINAL II czy III), lecz tu uwaga: Action Replay nie ma funkcji RENUMBER.

Aby zmienić numerację linii w programie należy uruchomić poniższy RENUMBER 64, następnie wczytać program do przenumerowania i skorzystać z jednej z podanych poniżej instrukcji.

SYS 49152,X,Y, gdzie
X — nr pierwszej linii,
Y — odległość kroku

lub

SYS 49152,X,Y,Z
X — nr pierwszej linii,
Y — długość kroku,
Z — renumeracja od tej linii programu.

MARIUSZ FERDYN

```
100 REM **********************
105 REM *   RENUMBER PLUS    *
110 REM *        BY          *
115 REM *     M.FERDYN       *
120 REM **********************
125 D=49152:B=63639
130 C=0:E=D
135 READ A$:IF A$="END" THEN 185
140 A1=ASC(LEFT$(A$,1)) AND 63
145 A2=ASC(RIGHT$(A$,1)) AND 63
150 IF A1>47 THEN 160
155 A1=A1+9:GOTO 165
160 A1=A1-48
165 IF A2>47 THEN A2=A2-48:GOTO 175
170 A2=A2+9
175 A=A1*16+A2:POKE D,A
180 D=D+1:C=C+A:GOTO 135
185 IF C<>B THEN PRINT "BLAD W LINI
    ACH DATA":STOP
190 PRINT:PRINT CHR$ (5):PRINT "REN
    UMBER PLUS"
195 PRINT "      BY"
200 PRINT "  M.FERDYN"
205 PRINT:PRINT "SYS 49152,X,Y"
210 PRINT "X - NR. PIERWSZEJ LINII"
215 PRINT "Y - KROK"
220 PRINT:PRINT "SYS 49152,X,Y,Z"
225 PRINT "X - NR. PIERWSZEJ LINII"
230 PRINT "Y - ODLEGLOSC KROKU"
235 PRINT "Z - RENUMERACJA OD TEGO
    WIERSZA":NEW
240 DATA 20,73,00,F0,40,20,6B,A9
245 DATA A5,14,A6,15,85,3D,86,3E
250 DATA 20,FD,AE,20,6B,A9,A5,14
255 DATA A6,15,85,3F,86,40,A0,00
260 DATA A9,2C,D1,7A,D0,11,20,FD
265 DATA AE,20,6B,A9,A5,14,A6,15
270 DATA 85,41,86,42,4C,54,C0,A0
275 DATA 02,B1,2B,85,41,C8,B1,2B
280 DATA 85,42,4C,61,C0,A9,0A,85
285 DATA 3D,85,3F,A9,00,85,3E,85
290 DATA 40,4C,37,C0,A5,3D,C5,41
295 DATA A5,3E,E5,42,B0,03,4C,48
300 DATA B2,20,FA,C1,A0,01,B1,43
305 DATA F0,38,A9,FF,85,14,85,15
310 DATA 20,B7,C1,20,8E,A6,4C,AE
315 DATA C0,A0,00,B1,43,AA,C8,B1
320 DATA 43,F0,1F,48,C8,A5,3D,91
325 DATA 43,C8,A5,3E,91,43,86,43
330 DATA 68,85,44,A5,3D,18,65,3F
335 DATA 85,3D,A5,3E,65,40,85,3E
340 DATA 90,D7,20,60,A6,4C,86,E3
345 DATA E6,7A,D0,02,E6,7B,A0,00
350 DATA B1,7A,D0,13,A0,02,B1,7A
355 DATA F0,BF,A5,7A,18,69,05,85
360 DATA 7A,90,EB,E6,7B,B0,E7,C9
365 DATA 22,D0,0B,20,73,00,C9,00
370 DATA F0,E2,C9,22,D0,F5,C9,89
375 DATA F0,17,C9,8D,F0,13,C9,A7
380 DATA F0,0F,C9,8[illegible] F0,0B,C9,CB
385 DATA D0,BE,20,73,00,C9,A4,D0
390 DATA BF,20,73,00,B0,BA,84,14
395 DATA 84,15,E9,2F,90,33,AA,A5
400 DATA 15,85,22,C9,19,B0,ED,A5
405 DATA 14,0A,26,22,0A,26,22,65
410 DATA 14,85,14,A5,22,65,15,85
415 DATA 15,06,14,26,15,8A,65,14
420 DATA 85,14,90,02,E6,15,C8,B1
425 DATA 7A,C9,20,F0,F9,C9,3A,90
430 DATA C9,A5,14,C5,41,A5,15,E5
435 DATA 42,90,3E,84,CE,20,B7,C1
440 DATA 38,A2,90,20,49,BC,20,DD
445 DATA BD,A0,FF,C8,B9,01,01,D0
450 DATA FA,A5,7A,85,5F,A5,7B,85
455 DATA 60,38,98,E5,CE,30,1C,F0
460 DATA 36,85,CE,A5,2D,85,5A,18
465 DATA 65,CE,85,58,A5,2E,85,5B
470 DATA 69,00,85,59,20,B8,A3,F0
475 DATA 1E,90,30,49,FF,A8,C8,A2
480 DATA 00,A5,5F,C5,2D,A5,60,E5
485 DATA 2E,B0,0C,B1,5F,81,5F,E6
490 DATA 5F,D0,EE,E6,60,D0,EA,A0
495 DATA 00,B9,01,01,F0,05,91,7A
500 DATA C8,D0,F6,20,FA,C1,20,73
505 DATA 00,90,FB,A0,00,C9,2C,D0
510 DATA 02,A9,89,AA,4C,B2,C0,A5
515 DATA 3D,A6,3E,85,63,86,62,A5
520 DATA 43,A6,44,85,22,86,23,A0
525 DATA 02,B1,22,C5,14,C8,B1,22
530 DATA E5,15,B0,25,A0,00,B1,22
535 DATA AA,C8,B1,22,86,22,85,23
540 DATA B1,22,F0,15,A5,63,65,3F
545 DATA 85,63,A5,62,65,40,85,62
550 DATA B0,04,C9,FA,90,D1,4C,48
555 DATA B2,60,20,33,A5,18,A5,22
560 DATA 69,02,85,2D,A5,23,69,00
565 DATA 85,2E,A5,41,A6,42,85,14
570 DATA 86,15,20,13,A6,A5,5F,85
575 DATA 43,86,44,60,END
```

LUPA dla C-64

Program powiększa znaki grafiki niskiej rozdzielczości ośmiokrotnie. Po uruchomieniu (i zapisaniu) wpisujemy, z którego zestawu znaków będziemy w danej chwili korzystać (1 — duże litery i znaki graficzne, 2 — małe i duże litery). Następnie wpisujemy znak, który ma być powiększony (również znaki w rewersie). Po dokonaniu wyboru zestawu znaków i samego znaku naciskamy klawisz RETURN. Znak, który odzwierciedla punkt „włączony" i „wyłączony" można zmienić (linie 170 i 175).

KAROL SKOWROŃSKI

```
100 rem *** lupa dla c-64 ***
105 rem *  karol skowronski *
110 rem *********************
115 :
120 poke 53280,0:poke 53281,0:pok
    e 646,12
125 printchr$(8):open 1,0:b$="zna
    k:"
130 a$="ktory zestaw znakow ? (1/
    2):"
135 printchr$(147)a$;:input#1,c$
140 print:printb$;:input#1,d$:k=0
145 if left$(c$,1)="2" then k=204
    8:print chr$(14)
150 print chr$(147)b$left$(d$,1)
155 a=peek(1029)*8+k+53248
160 poke 56334,0:poke 1,51:for x=
    0 to 7
165 print:p=peek(x+a):for y=7 to
    0 step-1
170 if p=(p or 2^y) then printchr
    $(42);:goto 180
175 print chr$(46);
180 next y,x:poke 1,55:poke 56334
    ,1
185 print spc(43)"nacisnij 'retur
    n'"
190 get x$:if x$<>chr$(13) then 1
    90
195 print chr$(142):goto 135
```

POLSKIE LITERY i C-64

Ten program napisałem w celu dołożenia własnej cegiełki do „wiecznie" żywego problemu polskich znaków i ich uzyskiwania za pośrednictwem klawiatury C-64.

Po wpisaniu programu polskie znaki otrzymamy wciskając:

Ł — SHIFT + L
Ń — SHIFT + N
Ó — SHIFT + O
Ą — SHIFT + A
Ę — SHIFT + E
Ś — SHIFT + S
Ć — SHIFT + C
Ż — SHIFT + Z.

Ze znaków tych możesz korzystać tylko w trybie dużych liter i znaków graficznych (Business Mode).

Program zajmuje około 750 bajtów i śmiało można go włączyć jako procedurę do własnego programu.

PIOTR PĘCHERZ

```
100 rem *** polskie  litery ***
105 rem ***  piotr pecherz  ***
110 rem ***  (c) c&a 1992   ***
115 :
120 poke 52,48:poke 56,48:clr
125 poke 56334,peek(56334)and254
130 poke 1,peek(1)and251
135 print "chwilka..."
140 for x=0 to 4095:poke x+12288
    ,peek(53248+x):next
145 poke 1,peek(1)or4
150 poke 56334,peek(56334)or1
155 poke 53272,(peek(53272)and24
    0)+12:clr
160 for x=12896 to 12903:read q:
    poke x,q:next
165 data 48,48,52,120,176,48,48,
    62
170 for x=12808 to 12815:read q:
    poke x,q:next
175 data 24,60,102,126,102,102,1
    02,3
180 for x=12840 to 12847:read q:
    poke x,q:next
185 data 126,96,96,120,96,96,126
    ,8
190 for x=12952 to 12959:read q:
    poke x,q:next
195 data 8,60,102,96,60,6,102,60
200 for x=12824 to 12831:read q:
    poke x,q:next
205 data 8,60,102,96,96,96,102,60
210 for x=13008 to 13015:read q:
    poke x,q:next
215 data 8,126,6,12,24,48,96,126
220 for x=12912 to 12919:read q:
    poke x,q:next
225 data 8,102,118,126,126,110,1
    02,102
230 for x=12920 to 12927:read q:
    poke x,q:next
235 data 8,60,102,102,102,102,10
    2,60
240 print chr$(147)"gotowe!"
```

BEEP 64

```
100 for t=49152 to 49233
105 read a:poke t,a:next
110 print "start - sys 49152"
115 :
119 end
120 data 173,143,002,141,046,003,
    173,144,002
125 data 141,047,003,120,169,029,
    141,143,002
130 data 169,192,141,144,002,088,
    096,234,234
135 data 234,234,165,203,201,064,
    240,043,169
140 data 015,141,024,212,169,000,
    141,005,212
145 data 141,006,212,169,001,141,
    000,212,169
150 data 255,141,001,212,169,033,
    141,004,212
155 data 162,003,160,255,136,208,
    253,202,208
160 data 248,169,000,141,004,212,
    108,046,003,000
```

Program symuluje tzw. procedurę BEEP, czyli powoduje, że przy każdym naciśnięciu klawisza słyszymy dźwięk. Program dodaje krótką podprocedurę do procedury normalnie wykonywanej podczas odczytu stanu klawiatury.

Wykonując POKE 49211,x możemy zmienić rodzaj fali generowanego dźwięku, natomiast POKE 49201,x i POKE 49206,x zmienia wysokość dźwięku.

MARCIN TUTAJ

DIGITALIZACJA i MAGNETOFON

Przeglądając stare numery czasopisma „Bajtek" znalazłem w jednym z nich ciekawy program. Otóż dzięki temu programowi można było digitalizować dźwięki z magnetofonu (digitalizowanie polega na zamianie dźwięku z postaci analogowej na cyfrową, możliwą do zapamiętania przez komputer). Niestety, program ten miał jedną zasadniczą wadę — był przeznaczony dla komputera Spectrum. Pomyślałem więc, że można by zrobić podobny program dla naszego małego Commodore. Już po kilkudziesięciu minutach wiedziałem, że nie będzie to takie proste.

Niestety, ze względu na budowę naszej maszynki, możliwa jest tylko digitalizacja jednobitowa. W praktyce oznacza to, że jakość będzie oględnie mówiąc, nie najlepsza. Kolejną wadą jest niewielka częstotliwość pracy procesora, co uniemożliwia uzyskanie zadowalających rezultatów. W efekcie, jedyne co udało mi się osiągnąć, to programik pozwalający na słuchanie „muzyki" przy pomocy Datasette. Program został napisany tak, aby dźwięk był jak najczystszy, lecz niestety, jeden bit i jeden MHz to o wiele za mało.

Po wpisaniu i uruchomieniu programu, ekran zostanie wygaszony. W tym momencie program jest już gotowy do pracy. Jedyne co pozostaje nam do zrobienia, to wciśnięcie klawisza PLAY w magnetofonie. Teraz możemy słuchać taśm z nagraną muzyką. Najlepiej brzmią rock'n'rolle, natomiast heavy metal jest jeszcze gorszy niż w oryginale. Życzę udanych eksperymentów i miłej zabawy.

J.B.

```
200 rem ***************************
205 rem * coded by jetboy/cavern *
210 rem ***************************
215 d=49152:b= 3206
220 c=0:e=d
225 read a$:if a$="end" then 275
230 a1=asc(left$(a$,1))and63
235 a2=asc(right$(a$,1))and63
240 if a1>47 then 250
245 a1=a1+9:goto 255
250 a1=a1-48
255 if a2>47 thena2=a2-48:goto265
260 a2=a2+9
265 a=a1*16+a2:poked,a
270 d=d+1:c=c+a:goto225
275 if c<>b then print"blad w lini
    ach data":stop
280 sys e
285 data 78,a9,2b,8d,11,d0,a9,07
290 data 85,01,a9,00,85,c0,a2,00
295 data ad,0d,dc,29,10,f0,02,a2
300 data 0f,8e,18,d4,4c,0e,c0,end
```

RAM-DYSK 64

Bardzo często zdarza się, że pracując nad jakimś programem zapisujesz sobie co jakiś czas kolejne wersje programu. Jeżeli jesteś posiadaczem magnetofonu, to rozumiem Twój ból w chwili, gdy chcesz zorientować się, która z wersji jest tą „ostateczną”... Poniższy program powinien Ci ułatwić życie w tym zakresie.

Jest to swoisty RAM-dysk. Niestety, jest on przydatny tylko przy układaniu programów w BASIC. Program ten po uruchomieniu instaluje się w obszarze od $cc00 do $cfff i dla swoich celów wykorzystuje obszar pamięci $cb00—$cbff oraz $d000—$ffff. Dodatkowo modyfikuje on wektor przerwań niemaskowalnych, tak więc procedura jest „niewidzialna” dla systemu. Oznacza to, że możemy uruchomić inne programy obsługujące bądź wykorzystujące do swoich celów przerwania IRQ. Jest to dopuszczalne pod warunkiem, że programy te NIE BĘDĄ korzystały one z pamięci leżącej powyżej adresu $cb00.

Wywołanie programu następuje po wciśnięciu klawisza RESTORE. Jeśli pracujesz w trybie ekranowym (bezpośrednim”, możliwe jest skopiowanie programu z pamięci do RAM-dysku lub z RAM-dysku do pamięci. Próba wywołania RAM-dysku podczas wykonywania programu przez komputer jest niedozwolona, co sygnalizowane jest odpowiednim komunikatem. Trzeba też pamiętać, że po wykonaniu jakiegoś programu musimy komputerowi przypomnieć, iż znajduje się on w trybie bezpośrednim; w tym celu wydaj mu dowolne polecenie w tym trybie (np. PRINT).

Do RAM-dysku możemy wpisywać programy, których długość nie przekracza 12 KB. Związane jest to z obszarem, jaki komputer wykorzystuje na nasz RAM-dysk — jest to pamięć „leżącą pod” Kernalem czyli obszar od adresu $d000 do $ffff. Próba zapisu programu dłuższego niż 12 KB zakończy się wyświetleniem odpowiedniego komunikatu.

W RAM-dysku może znajdować się tylko jeden program. Próba wpisania do RAM-dysku jakiegokolwiek nowego programu powoduje nieodwracalne skasowanie aktualnej zawartości naszej dodatkowej pamięci.

JB

```
1000 rem *************************
1010 rem *         ramdisk       *
1011 rem *   by jetboy/parados   *
1020 rem *************************
1030 d=52224:b= 79735
1040 c=0:e=d
1050 read a$:if a$="end" then 1150
1060 a1=asc(left$(a$,1))and63
1070 a2=asc(right$(a$,1))and63
1080 if a1>47 then 1100
1090 a1=a1+9:goto 1110
1100 a1=a1-48
1110 if a2>47 then a2=a2-48:goto
     1130
1120 a2=a2+9
1130 a=a1*16+a2:poke d,a
1140 d=d+1:c=c+a:goto 1050
1150 if c<>b then print "blad w 1
     iniach data":stop
1160 sys e
1170 data 78,a2,cc,a0,28,8c,18,03
1180 data 8e,19,03,a2,34,86,01,a2
1190 data d0,8e,19,cc,a2,00,8a,9d
1200 data 00,d0,e8,d0,fa,ee,19,cc
1210 data d0,f5,a2,37,86,01,58,60
1220 data 08,48,8a,48,98,48,20,38
1230 data cc,68,a8,68,aa,68,28,40
1240 data ae,6f,cd,d0,23,a2,ff,8e
1250 data 6f,cd,a6,3a,e0,ff,d0,19
1260 data a5,01,8d,70,cd,20,8f,cc
1270 data 20,b1,cc,20,a3,cc,ad,70
1280 data cd,85,01,a2,00,8e,6f,cd
1290 data 60,a2,00,bd,00,04,9d,00
1300 data cb,bd,77,cd,9d,00,04,e8
1310 data e0,c8,d0,ef,20,87,cc,a2
1320 data 00,bd,00,cb,9d,00,04,e8
1330 data e0,c8,d0,f5,4c,5b,cc,ad
1340 data 01,dc,c9,ef,d0,f9,60,a2
1350 data 00,bd,00,04,9d,00,cb,bd
1360 data 3f,ce,9d,00,04,e8,e0,f0
1370 data d0,ef,60,a2,00,bd,00,cb
1380 data 9d,00,04,e8,e0,f0,d0,f5
1390 data 60,ad,01,dc,c9,fe,f0,0c
1400 data c9,f7,f0,4b,c9,ef,f0,03
1410 data 4c,b1,cc,60,a5,2e,18,e9
1420 data 37,10,03,4c,e1,cc,a2,00
1430 data bd,2f,cf,9d,00,04,e8,e0
1440 data c8,d0,f5,20,87,cc,4c,c3
1450 data cc,a9,34,85,01,20,35,cd
1460 data a0,00,b1,fe,91,fc,c8,d0
1470 data f9,e6,ff,e6,fd,d0,f3,a5
1480 data 2d,8d,75,cd,a5,2e,8d,76
1490 data cd,20,5a,cd,4c,c3,cc,a9
1500 data 34,85,01,20,35,cd,a0,00
1510 data b1,fc,91,fe,c8,d0,f9,e6
1520 data ff,e6,fd,d0,f3,ad,75,cd
1530 data 85,2d,85,2f,85,31,ad,76
1540 data cd,85,2e,85,30,85,32,20
1550 data 5a,cd,4c,c3,cc,a5,fc,8d
1560 data 71,cd,a5,fd,8d,72,cd,a5
1570 data fe,8d,73,cd,a5,ff,8d,74
1580 data cd,a9,00,85,fc,a9,d0,85
1590 data fd,a9,00,85,fe,a9,08,85
1600 data ff,60,ad,71,cd,85,fc,ad
1610 data 72,cd,85,fd,ad,73,cd,85
1620 data fe,ad,74,cd,85,ff,60,00
1630 data 00,00,d0,00,00,01,08,70
1640 data 40,40,40,40,40,40,40,40
1650 data 40,40,40,40,40,40,40,40
1660 data 40,40,40,40,40,40,40,40
1670 data 40,40,40,40,40,40,40,40
1680 data 40,40,40,40,40,40,6e,5d
1690 data 20,20,20,20,14,0f,20,0e
1700 data 09,05,20,0a,05,13,14,20
1710 data 14,12,19,02,20,02,05,1a
1720 data 10,0f,13,12,05,04,0e,09
1730 data 21,21,20,20,20,20,5d,5d
1740 data 20,20,20,20,20,20,20,20
1750 data 20,20,20,20,20,20,20,20
1760 data 20,20,20,20,20,20,20,20
1770 data 20,20,20,20,20,20,20,20
1780 data 20,20,20,20,20,20,5d,5d
1790 data 13,10,01,03,0a,01,3d,03
1800 data 01,0e,03,05,0c,20,20,20
1810 data 20,20,20,20,20,20,20,20
1820 data 20,13,10,01,03,0a,01,3d
1830 data 03,01,0e,03,05,0c,5d,6d
1840 data 40,40,40,40,40,40,40,40
1850 data 40,40,40,40,40,40,40,40
1860 data 40,40,40,40,40,40,40,40
1870 data 40,40,40,40,40,40,40,40
1880 data 40,40,40,40,40,40,7d,70
1890 data 40,40,40,40,40,40,40,40
1900 data 40,40,40,40,40,40,40,40
1910 data 40,40,40,40,40,40,40,40
1920 data 40,40,40,40,40,40,40,40
1930 data 40,40,40,40,40,40,6e,5d
1940 data 12,01,0d,20,04,09,13,03
1950 data 20,16,30,2e,39,20,3a,28
1960 data 03,29,03,0f,0d,0d,0f,04
1970 data 0f,12,05,26,01,0d,09,07
1980 data 01,20,31,39,39,32,5d,5d
1990 data 2d,2d,2d,2d,2d,2d,2d,2d
2000 data 2d,2d,2d,2d,2d,2d,2d,2d
2010 data 2d,2d,2d,2d,2d,2d,2d,2d
2020 data 2d,2d,2d,2d,2d,2d,2d,2d
2030 data 2d,2d,2d,2d,2d,2d,5d,5d
2040 data 31,3d,10,12,0f,07,12,01
2050 data 0d,3d,3e,12,01,0d,04,09
2060 data 13,03,20,3a,20,13,10,01
2070 data 03,05,3d,10,0f,17,12,0f
2080 data 14,20,04,0f,20,20,5d,5d
2090 data 32,3d,10,12,0f,07,12,01
2100 data 0d,3c,3d,12,01,0d,04,09
2110 data 13,03,20,3a,20,20,20,20
2120 data 20,20,09,0e,14,05,12,10
2130 data 12,05,14,05,12,01,5d,6d
2140 data 40,40,40,40,40,40,40,40
2150 data 40,40,40,40,40,40,40,40
2160 data 40,40,40,40,40,40,40,40
2170 data 40,40,40,40,40,40,40,40
2180 data 40,40,40,40,40,40,7d,70
2190 data 40,40,40,40,40,40,40,40
2200 data 40,40,40,40,40,40,40,40
2210 data 40,40,40,40,40,40,40,40
2220 data 40,40,40,40,40,40,40,40
2230 data 40,40,40,40,40,40,6e,5d
2240 data 20,10,12,0f,07,12,01,0d
2250 data 20,1a,01,20,04,0c,15,07
2260 data 09,20,2d,20,1a,01,10,09
2270 data 13,20,0e,09,05,0d,0f,1a
2280 data 0c,09,17,19,21,20,5d,5d
2290 data 20,20,20,20,20,20,20,20
2300 data 20,20,20,20,20,20,20,20
2310 data 20,20,20,20,20,20,20,20
2320 data 20,20,20,20,20,20,20,20
2330 data 20,20,20,20,20,20,5d,5d
2340 data 20,13,10,01,03,05,3d,03
2350 data 01,0e,03,05,0c,20,20,20
2360 data 20,20,20,20,20,20,20,20
2370 data 20,13,10,01,03,05,3d,03
2380 data 01,0e,03,05,0c,20,5d,6d
2390 data 40,40,40,40,40,40,40,40
2400 data 40,40,40,40,40,40,40,40
2410 data 40,40,40,40,40,40,40,40
2420 data 40,40,40,40,40,40,40,40
2430 data 40,40,40,40,40,40,7d,end
```

LIST-STOP

Przedstawiam króciutki, bo liczący zaledwie 23 bajty program w języku maszynowym przeznaczony dla C-64. Pozwala on na zatrzymanie listowania programów przez naciśnięcie klawisza SHIFT (lub SHIFT LOCK). Po zwolnieniu klawisza listowanie jest wznawiane. Działanie programu jest identyczne w przypadku, gdy urządzeniem wyjściowym jest drukarka lub stacja dysków.

Program zajmuje obszar \$C000–\$C016 i jest w pełni relokowalny. Można go zawsze uruchomić (np. po RESET) instrukcją SYS 49152.

GRZEGORZ NOWIK

```
100 rem ********************
105 rem *     list-stop    *
110 rem * g.nowik (c) 1992 *
115 rem ********************
120 :
125 adr=49152
130 for i=adr to adr+22:read q:po
    ke i,q:next
135 sys adr
140 printchr$(147)"gotowe!"
145 data 162,011,160,192,142,006,
    003,140,007,003,096,072,173,
150 data 141,002,201,001,240,249,
    104,076,026,167
```


FILE PROTECTOR C-16/116/+4

Ostatnio bardzo popularne stało się kodowanie programów operacjami logicznymi EOR na hasło. Tym razem chciałbym przedstawić program służący do tego celu dla komputerów rodziny C-16/116/PLUS/4.

Instrukcja użytkowania zawarta jest w samym programie. Odgadnięcie hasła jest praktycznie niemożliwe, gdyż przy haśle 4 literowym istnieje ponad 3000 kombinacji znaków. Natomiast przy haśle składającym się z 20 liter mamy $3.11*10^{20}$ możliwości.

Uwaga:
Maksymalna długość hasła nie może przekraczać 47 znaków.

MARIUSZ FERDYN

```
200 rem ******************************
205 rem *      file protector        *
210 rem *            by              *
215 rem *         m.ferdyn           *
220 rem ******************************
225 poke 216,peek(55):poke 217,peek(56)
230 poke 56,(peek(56))-10:clr
235 d=320:b=1821:c=0
240 read a$:if a$="end" then 255
245 a=dec(a$):poke d,a
250 d=d+1:c=c+a:goto 240
255 if c<>b then print "blad w liniach d
    ata blok 1":stop
260 rem **** data blok 1 ****
265 data 0d,0d,57,50,52,4f,57,41
270 data 44,5a,20,48,41,53,4c,4f
275 data 3a,00,0d,0d,43,4f,4d,50
280 data 4c,45,54,45,20,21,00,end
285 rem **** data blok 2 ****
290 data 162,048,169,000,157,255,001,202
295 data 208,250,162,000,189,064,001,201
300 data 000,240,006,032,210,255,232,208
305 data 243,162,000,032,207,255,201,013
310 data 240,006,157,000,002,232,208,243
315 data 224,000,240,075,142,059,001,165
320 data 044,133,219,133,217,165,043,133
325 data 218,133,216,160,000,177,218,192
330 data 000,240,073,160,000,145,216,165
335 data 218,197,045,240,032,165,218,201
340 data 255,240,008,230,218,230,216,192
345 data 000,240,224,165,219,201,207,240
350 data 022,230,218,230,219,230,216,230
355 data 217,192,000,240,206,165,219,197
360 data 046,240,004,192,000,240,214,162
365 data 000,189,082,001,201,000,240,006
370 data 032,210,255,232,208,243,160,000
375 data 192,000,240,044,160,000,089,000
380 data 002,162,000,141,058,001,189,000
385 data 002,024,125,001,002,157,000,002
390 data 232,236,059,001,208,240,173,058
395 data 001,200,204,059,001,208,223,141
400 data 017,255,160,000,192,000,240,139
405 data 096,160,000,177,043,145,045,169
410 data 000,145,043,200,177,043,145,045
415 data 169,000,145,043,165,045,024,105
420 data 002,133,045,176,001,096,230,046
425 data 096,165,045,024,105,254,133,045
430 data 176,002,198,046,160,000,177,045
435 data 145,043,169,000,145,045,200,177
440 data 045,145,043,169,000,145,045,096
445 data 189
450 mb=peek(216):sb=peek(217)
455 ts=(sb*256)+mb-250
460 de=ts+248
465 for i=ts to de
470 read p:poke i,p:z=z+p:next
475 if s<> 31716 then print "blad w lini
    ach data blok 2":stop
480 scnclr:print "             file prot
    ector":rem 13 spacji
485 print "                   by":rem 19
    spacji
490 print "                m.ferdyn":rem
    16 spacji
495 print:print "1. wgraj program do zak
    odowania":print "  lub odkodowania."
500 print "2. w celu zakodowania program
    u wykonaj:"
505 print "      sys";ts;"+ [return] [f4
    ] - key":key 4,"sys"+str$(ts)+chr$(1
    3)
510 print "      po wpisaniu hasla i wyk
    onaniu"
515 print "      procedury kodowania wyk
    onaj:"
520 print "      sys";ts+185;"+ [return]
    [f5]"
525 key 5,"sys"+str$(ts)+"+185"+chr$(13)
530 print "2. w celu odkodowania program
    u nacisnij"
535 print "      klawisz f1"
540 key 1,"sys"+str$(ts)+"+217:sys"+str$
    (ts)+chr$ (13)
545 print "3. gdy pojawi sie kursor prog
    ram"
550 print "  jest zakodowany lub odkodo
    wany"
555 print "  i mozesz go zgrac na nosni
    k"
560 print "  lub uruchomic."
565 ts=ts-1:sb=int(ts/256):mb=(ts-(sb*25
    6))
570 poke 55,mb:poke 56,sb:clr
575 e=peek(56)*256+peek(55)
580 s=peek(44)*256+peek(43)
585 print:print e-s;" bytes free":clr:new
```

muzyka i C-128

Omówione w ubiegłym miesiącu instrukcje PLAY, VOL i TEMPO pozwalają dowolnie kształtować takie elementy muzyczne, jak melodyka, dynamika, tempo czy rytm. Jednak abyśmy mogli w pełni panować nad materią muzyczną, potrzebna jest nam jeszcze możliwość modyfikacji barwy (timbre) brzmienia oraz artykulacji. Osobom nie znającym tych pojęć proponuję małą powtórkę z teorii muzyki.

Jak wiadomo, brzmienie instrumentów muzycznych, czy ogólniej: przedmiotów pobudzonych do drgań, ma znamiona indywidualne. Dwie nuty o identycznej wysokości, wykonane najpierw na puzonie, potem na skrzypcach, zabrzmią zupełnie inaczej. To, co je różni, to właśnie barwa brzmienia, lub używając terminologii nauk ścisłych, kształt fali dźwiękowej.

Artykulacja zaś jest to sposób wydobycia dźwięku, np. *staccato*, *legato* czy *portamento*. Dźwięki grane *staccato* są bardzo krótkie, zawsze słyszy się między nimi „dziury", tj. pauzy. Przeciwieństwem *staccato* jest *legato*: dźwięki przechodzą płynnie jeden w drugi bez żadnych przerw. Artykulacja wiąże się ściśle z czynnikiem czasu i pozwala zróżnicować tony nawet, jeśli są one tożsame pod względem barwy i wysokości. Odpowiednikiem artykulacji z punktu widzenia fizyki są zmiany natężenia (amplitudy) dźwięku.

Nieograniczoną praktycznie kontrolę nad barwą brzmienia i artykulacją dają użytkownikom C-128 instrukcje ENVELOPE i FILTER. Pozwalają one wydobyć z układu SID nawet najbardziej wyrafinowane niuanse brzmieniowe, a w ręku doświadczonego programisty stają się, wraz z innymi instrukcjami muzycznymi BASIC V7.0, idealnym narzędziem kompozytorskim.

ENVELOPE

Format:

ENVELOPE no,czn,czo,czw,czz,kf,si

no — numer obwiedni (0–9)
czn — czas narastania dźwięku (0–15)
czo — czas opadania dźwięku (0–15)
czw — czas wybrzmiewania dźwięku (0–15)
czz — czas zaniku dźwięku (0–15)
kf — kształt fali: 0 — sinusoidalny
1 — trójkątny
2 — prostokątny
3 — szum
4 — modulacja dookrężna
si — szerokość impulsu dla fali prostokątnej (0–4095).

W poprzednim odcinku, przy okazji omawiania instrukcji PLAY, była mowa o dziesięciu gotowych obwiedniach, zawartych w ROM. Jeżeli w programie nie użyjemy instrukcji ENVELOPE, to i tak za pomocą PLAY możemy wybrać jedną spośród dziesięciu obwiedni. ENVELOPE umożliwia jednak tworzenie całkiem nowych obwiedni, względnie modyfikację istniejących. Obwiednię, w której chcemy dokonać zmian, wybieramy parametrem no.

Kolejne cztery parametry (czn, czo, czw, czz) pozwalają modulować amplitudę fali dźwiękowej, czyli — innymi słowy — kształtować artykulację. Przyjęto podział czasu trwania dźwięku na cztery etapy, fazy. Są to: narastanie, opadanie, wybrzmiewanie i zanik (attack, decay, sustain i release — ADSR). Graficznie przedstawiono to na rysunku 1.

□ Narastanie (attack)

Jest to czas, w którym dźwięk osiąga maksymalne natężenie. Zauważcie, że w instrumentach, których brzmienie ma charakter perkusyjny (fortepian, perkusja, gitara, ksylofon), narastanie ustawione jest na zero. Oznacza to, że dźwięk osiąga swoje szczytowe natężenie natychmiast. W innych instrumentach (obwiedniach), jak np. akordeon, trąbka czy flet, dźwięk uzyskuje największą amplitudę dopiero po jakimś czasie. Im większą wartość przypiszemy parametrowi czn, tym łagodniej dźwięk osiąga szczytowe natężenie.

□ Opadanie (decay)

Czas, w którym natężenie dźwięku zmniejsza się do poziomu ustalonego. Im większą wartość przyjmie parametr czo, tym dłużej trwać będzie faza opadania.

☐ Wybrzmiewanie (sustain)

Czas, w którym natężenie dźwięku utrzymuje się na ustalonym poziomie. Jeśli parametr czw przyjmie wartość 0, dźwięk zaniknie całkowicie już w fazie wybrzmiewania. Jeśli zaś wybrzmiewanie ustawimy na 15 (maksymalna wartość), to parametr czo musi przyjąć wartość 0. W przeciwnym razie wystąpią nieoczekiwane efekty.

☐ Zanik (release)

Ostatnia faza trwania dźwięku, na koniec której jego amplituda osiąga wartość zero. Parametrowi określającemu fazę zaniku (czz) należy przypisywać raczej małe wartości — długi czas trwania fazy zaniku powoduje czasem, że dźwięki brzmią pomimo ich wyłączenia.

Doświadczenie w ustalaniu czasu trwania poszczególnych faz dźwięku najlepiej zdobyć metodą prób i błędów. Pomoże Wam w tym program z listingu 1, który służy do tworzenia własnych obwiedni. Wyniki eksperymentów są słyszalne w postaci zmienionego brzmienia gamy C-dur. Jako podstawę do porównań wybrałem obwiednię nr 0 (fortepian), ale oczywiście można ją łatwo zmienić (linie 110, 115 i 185).

Kolejnym parametrem instrukcji ENVELOPE jest kf. Umożliwia on wybór kształtu fali, dzięki czemu można wpływać na barwę brzmienia. Ciekawe efekty, jak np. odgłosy imitujące dźwięk gongu, dzwonka, uzyskujemy przypisując kf wartość 4. Zabieg ten powoduje aktywację modulacji dookrężnej, ale — uwaga! — tylko wtedy, gdy pierwszemu generatorowi będzie przypisana fala trójkątna.

Jeśli wybierzemy falę prostokątną, musimy dodatkowo określić szerokość impulsu (si). Wyraża się ona wzorem:

Szerokość impulsu [ms] =INT (X/40.95)

gdzie X to wartość przypisana parametrowi si. A jaki efekt powoduje zmiana wartości parametru si? Najlepiej przekonajcie się sami. Wpiszcie poniższy programik:

```
10 ENVELOPE 0,0,9,0,0,2,...
20 PLAY "V1T0O4CDEFGABO5C"
```

a w miejsce kropek wstawcie kolejno liczby 20, 100, 500, 2500 i 4000. Słychać wyraźnie ogromną zmianę barwy dźwięku, prawda?

Na koniec omówienia instrukcji ENVELOPE zobaczmy, jak przedstawiają się parametry obwiedni zawartych w ROM (powinno się je znać, choćby w celach porównawczych):

no	czn	czo	czw	czz	kf	si	instrument
0	0	9	0	0	2	1536	fortepian
1	12	0	12	0	1		akordeon
2	0	0	15	0	0		organki
3	0	5	5	0	3		perkusja
4	9	4	4	0	0		flet
5	0	9	2	1	1		gitara
6	0	9	0	0	2	512	klawesyn
7	0	9	9	0	2	2048	organy
8	8	9	4	1	2	512	trąbka
9	0	9	0	0	0		ksylofon

Trzeba przyznać, że ci, którzy ustalali powyższe parametry, nie wysili się zbytnio, wskutek czego niektóre obwiednie niewiele mają wspólnego z rzeczywistym brzmieniem danego instrumentu. Brzmienie gitary o wiele lepiej naśladuje obwiednia o następujących parametrach: ENVELOPE 5,0,9,0,0,2,50, a brzmienie klawesynu należałoby raczej zdefiniować obwiednią ENVELOPE 6,0,9,2,1,2,35.

Inaczej przedstawia się problem uzyskania brzmienia takich instrumentów jak fortepian, flet czy trąbka. Sama instrukcja ENVELOPE do tego nie wystarczy. Aby osiągnąć zadowalające efekty, trzeba skorzystać z "usług" kolejnego polecenia muzycznego BASIC V7.0 — FILTER.

FILTER

Format:

Filter cz,fd,fp,fg,r

cz — częstotliwość (0–2047)
fd — filtr dolnoprzepustowy (1 = włączony, 0 = wyłączony)
fp — filtr pasmowy (1 = włączony, 0 = wyłączony)
fg — filtr górnoprzepustowy (1 = włączony, 0 = wyłączony)
r — rezonans (0–15).

Filtrowanie polega na eliminacji pochodnych harmonicznych danego dźwięku, tzw. alikwotów. Muzycy mogą to robić za pomocą różnego rodzaju tłumików. Na pewno słyszeliście nie raz brzmienie trąbki bez tłumika i z tłumikiem. Różnica jest kolosalna. Układ SID dysponuje trzema filtrami, które działają właśnie jak tłumiki. Filtr dolnoprzepustowy blokuje częstotliwości harmoniczne wyższe od częstotliwości zadanej w parametrze cz, filtr górnoprzepustowy blokuje częstotliwości niższe, a filtr pasmowy — niższe i wyższe, ale tylko w wąskim zakresie.

Filtry możemy włączać i wyłączać pojedyńczo lub w kombinacjach i oczywiście należy pamiętać, aby instrukcja FILTER zawsze poprzedzała instrukcję PLAY. Ponieważ SID posiada tylko jeden układ filtrujący, raz zdefiniowane parametry filtrowania obowiązują wszystkie głosy, aż do czasu, gdy zostaną podane nowe wartości parametrów. Poszczególne głosy mogą jednak być filtrowane oddzielnie. W tym celu należy jedynie prawidłowo posługiwać się wyrażeniem Xn w instrukcji PLAY.

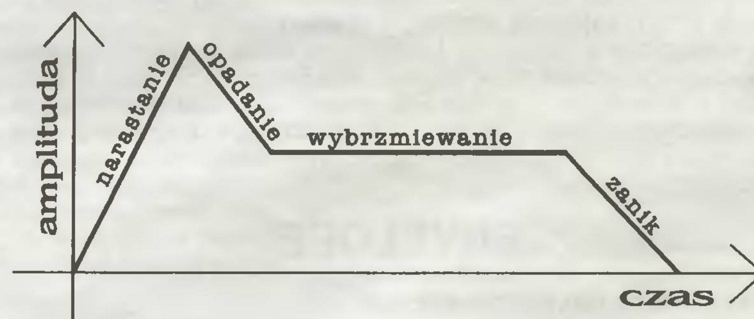
Parametr cz odnosi się do pochodnych harmonicznych i wyznacza ich częstotliwości poddawane filtrowaniu. Wartości, jakie może on przyjąć (0–2047), odpowiadają mniej więcej obszarowi częstotliwości od 30 do 12000 Hz. Dlaczego mniej więcej? Bo ilość i częstotliwość pochodnych harmonicznych zmieniają się w zależności od wysokości i barwy dźwięku.

Ostatni parametr instrukcji FILTER może przyjąć wartości od 0 (brak rezonansu) do 15 (rezonans najsilniejszy). A co to jest rezonans? Jest to jakby akcentowanie częstotliwości harmonicznych, nieco wyższych i nieco niższych od częstotliwości określonej parametrem cz.

Jak widzicie, instrukcja FILTER nie jest taką sobie zwykłą instrukcją. Zmiany uzyskane wskutek jej stosowania mogą być często bardzo subtelne, prawie niezauważalne, w konsekwencji przyczyniają się jednak do kreacji wysublimowanych, unikalnych brzmień.

Umiejętne posługiwanie się poleceniem FILTER wymaga właściwie tylko dwóch rzeczy: dobrego słuchu i sporej praktyki. To pierwsze można mieć lub nie, ale praktyka jest w zasięgu każdego. A pierwsze ćwiczenie mogę Wam zadać już teraz: wykorzystując obwiednię z ROM nr 8 oraz odpowiednio "nastrajając" filtry spróbujcie uzyskać prawdziwy dźwięk trąbki, a nie marny falsyfikat.

A teraz uwaga! Jeśli jesteście akurat w trakcie pisania w języku BASIC własnej gry i potrzebujecie efektów specjalnych typu odgłos kroków, odgłos spadania z gałęzi, seria z cekaemu czy tremolo na werblu w wykonaniu plutonu egzekucyjnego, koniecznie przeczytajcie te kilka słów o następnej i ostatniej instrukcji muzycznej C-128 — SOUND.



SOUND

Instrukcja ta służy w zasadzie do efektów specjalnych, choć można też wykorzystać ją do prawdziwego muzykowania, o czym przekona Was zamieszczony listing nr 2. Format tej instrukcji to:

SOUND ng,cz,czt,t,czm,pcz,kf,si

ng — numer generatora (1–3)
cz — częstotliwość (0–65535)
czt — czas trwania dźwięku określony wzorem:
czt/60 = czas w sekundach
t — tryb generowania dźwięku (0–2);
0 = generowany dźwięk będzie przybierał coraz wyższe częstotliwości w zakresie ustalonym parametrami cz i czm i z przyrostem określonym parametrem pcz;
1 = jak wyżej, ale częstotliwości będą coraz niższe;
2 = częstotliwości będą raz coraz niższe, raz coraz wyższe;
czm — częstotliwość minimalna (0–65535)
pcz — przyrost częstotliwości (0–32767); ustalony na 0 spowoduje, że częstotliwości dźwięków generowanych wg parametrów t, cz i czm zmieniać się będą płynnie;
kf — kształt fali:
0 — sinusoidalny
1 — trójkątny
2 — prostokątny
3 — szum
si — szerokość impulsu (0–4095; tylko dla fali prostokątnej)

Cóż, i to by było na tyle. Chociaż nie, zademonstruję Wam jeszcze kilka efektów:

SOUND 1,30000,120 — pojedyńczy dźwięk przez 2 sekundy
SOUND 1,30000,240,1,0,100,1 — spadanie
SOUND 1,10000,300,2,50,100,0 — spadanie i wznoszenie się
SOUND 1,21000,120,2,11000,5000,0 — telefon
SOUND 1,15000,120,2,5000,10000,3 — seria z CKM

Jeszcze lepszą zabawę możecie mieć generując dźwięki losowo. Służy do tego poniższy programik:

```
100 A  = INT (RND (1) *3) +1
105 B  = INT (RND (1) *65535)
110 C  = INT (RND (1) *240)
115 D  = INT (RND (1) *3)
120 E  = INT (RND (1) *65535)
125 F  = INT (RND (1) *32767)
130 G  = INT (RND (1) *4)
135 H  = INT (RND (1) *4095)
140 SOUND A,B,C,D,E,F,G,H
145 SLEEP 3
150 GOTO 100
```

Jak już wspomniałem, SOUND można też użyć w "normalnej muzyce". W listingu nr 2, instrukcja ta jest odpowiedzialna za tremolo, bez którego słynna melodia z "Ojca chrzestnego" jest przecież nie do pomyślenia.

CHRISTIAN GRZENKOWICZ

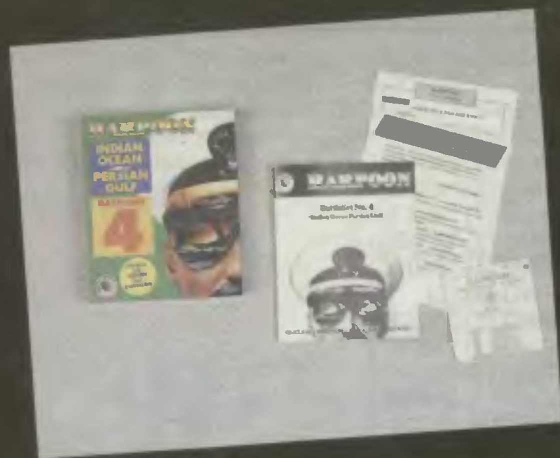
LISTING 1

```
 90 REM LISTING #1
 99 :
100 VOL 15:RV$=CHR$(18)
105 SCNCLR:A$="QO4CDEFGABO5CRQ"
110 PRINT RV$;"DANE GOTOWEJ OBWIE
    DNI NR 0:":PRINT
115 PRINT"ENVELOPE 0,0,9,0,0,2,15
    36":PRINT
120 PRINT "PODAJ SWOJE DANE:":PRI
    NT
125 INPUT "NARASTANIE";N
130 INPUT "OPADANIE";O
135 INPUT "WYBRZMIEWANIE";W
140 INPUT "ZANIK";Z
145 INPUT "RODZAJ FALI";RF
150 IF RF<>2 THEN 160
155 INPUT "SZEROKOSC PULSU";SP
160 PRINT:PRINT "TAK BRZMI NORMAL
    NIE..."
165 PLAY "V1T0"+A$
170 PRINT:PRINT "A TAK Z TWOIMI D
    ANYMI..."
175 ENVELOPE 0,N,O,W,Z,RF,SP
180 PLAY "V1T0"+A$
185 ENVELOPE 0,0,9,0,0,2,1536
190 GETKEY A$
195 GOTO 105
```

LISTING 2

```
 90 REM LISTING #2
 99 :
100 TEMPO 10
105 ENVELOPE 1,12,0,12,0,2,500
110 ENVELOPE 8,0,12,2,0,2,1800
115 FILTER 2000,1,0,0,15
120 PLAY "V2T1X1V3T8X1"
125 PLAY "V2O4QEAO5CM"
130 SOUND 1,9630,120,2,8080,1550,0
135 PLAY "V3O2WAV2O4QBAO5CO4A"
140 SOUND 1,6480,120,2,4500,1980,0
145 PLAY "V3O2WDV2O4QBAFG"
150 SOUND 1,6200,120,2,4700,1500,0
155 PLAY "V3O2WAV2O4WE"
160 PLAY "QRO5QEAO6CM"
165 SOUND 1,9630,240,2,8080,1550,0
170 PLAY "V3O2WAV2O5QBAO6CO5A"
175 PLAY "V3O1WAV2O5QBAFE"
180 SOUND 1,13000,120,2,11000,2000,0
185 PLAY "V3O2WDV2O5WD"
190 PLAY "QRV2O5DF#GM"
195 SOUND 1,13000,240,2,11000,2000,0
200 PLAY "V2O5WBV3O2HDF"
205 PLAY "V3O2H#GV2QRO5QDF#G"
210 SOUND 1,12250,240,2,9630,2620,0
215 PLAY "V2O5WAV3O2HCE"
220 PLAY "V3O2HAV2QRO4AO5CG"
225 SOUND 1,6200,240,2,4000,2200,0
230 PLAY "V3O3WEV2O5QFEGF"
235 PLAY "V3O2WEV2O5QFEEO4#G"
240 SOUND 1,9630,120,2,8080,1550,0
245 PLAY "V3O2HAV2O4WAV3O3QCEWA"
```

GRY



GRY



DYSTRYBUTOREM PROGRAMOW „STEEL EMPIRE", „POWERMONGER" JEST FIRMA IPS COMPUTER GROUP, WARSZAWA, UL. OKRĘŻNA 3, TEL. 642-2766, 642-2768.

STEEL EMPIRE

Już sama okładka, przedstawiająca wyłaniające się z ciemności armie robotów i lecące w naszą stronę rakiety balistyczne skutecznie zniechęca... pacyfistów. Dla nich ta gra bowiem nie jest przeznaczona. Natomiast wszyscy, którym miły jest widok eksplodujących cyborgów przeciwnika gra może wydać się ciekawa.

Jak podają producenci, pięć potęg wojskowo-przemysłowych walczy o dominację nad planetą Orion mającą duże znaczenie strategiczne. Wszyscy gracze zaczynają od jednego skrawka terenu, aby — podbijając po kolei pozostałe 71 kawały gruntu — zawładnąć w końcu całą planetą, co, jak wiemy jest marzeniem wszystkich obłąkanych przywódców przeróżnych potęg wojskowo-przemysłowych (i to nie tylko na powstałej w chorej wyobraźni autorów planecie Orion).

Aby jednak pobić przeciwników potrzeba nie tylko dobrych chęci. Niezbędne są jeszcze wyimaginowane pieniądze, za które możemy sobie postawić wyimaginowane fabryki, produkujące na zamówienie wyimaginowane cyborgi. Cyborgi wytwarzane są w dziewięciu rodzajach — od przypominającego pchełkę Merkurego do, zaiste tytanicznego, Tytana. Jako, że na planecie istnieją różne strefy klimatyczne — od arktycznej do pustynnej — każdy robot ma dość ściśle określone strefy, w których sprawuje się najlepiej. Jak w większości znanych mi gier wojennych, pieniądze zdobywa się dzięki okupacji poszczególnych pól. Każde pole ma swą ściśle określoną wartość, która zależy od jego rodzaju a także ogólnego bogactwa planety. Ciekawą innowacją w tej grze jest wprowadzenie stolic. W każdym z podbitych terytoriów do ściągania dla nas pieniędzy służy przez nas zakładany i ściśle kontrolowany rząd (czyżby marionetkowy?).

Dość dobrym pomysłem jest połączenie elementów gry strategicznej i zręcznościowej. Jeśli jednak ktoś nie przepada za rąbanką i łamaniem joysticka — zawsze może wybrać sobie grę czysto strategiczną. Można też wybrać opcję zręcznościową, w której efektywność gracza jest oceniana na podstawie tonażu zdemolowanych maszyn wroga. Daje to graczowi większą różnorodność i większą możliwość wyboru. Podczas gry możemy słuchać sobie syntezatorka mowy, który nas wyśmiewa lub chwali, w zależności od postępów ofensywy. Każde pole możemy — jeśli uznamy to za niezbędne — ufortyfikować, co wydatnie wzmacnia jego obronność.

Autorzy gry nie ustrzegli się jednak kilku potknięć. Nie są to błędy rażące, uniemożliwiające dobrą zabawę, ale jednak trochę ją psują. Np. ikony, które przecież w założeniu twórców miały ułatwić komunikację z komputerem są dość nieczytelne i wspomagać nas musi dołączona do gry mapa świata z objaśnieniami tychże ikon. Dość nieprzyjemne jest zabezpieczenie przed kopiowaniem, które prosi o podanie określonego słowa z instrukcji obsługi. Czasem zdarza się niestety, że musimy szukać akapitu dziesiątego na stronie, która ma tych akapitów dziewięć. Podczas gry, jeśli już coś zadecydujemy, nie mamy żadnej możliwości odwołania naszych rozkazów (być może ma to być jakiś wątek dydaktyczny tej gry — ma wpoić odpowiedzialność za czyny czy głębsze zastanawianie się nad kolejnym ruchem). Zaś w wersji niemieckiej gry nie mamy wcale „umlautów" i niektóre słowa tracą sens.

W sumie „Steel Empire" należy moim zdaniem do przeciętnych gier strategicznych; nie znaczy to jednak wcale, że gra jest zła. Po prostu nie udało się jej mnie oczarować.

AMPER

POWERMONGER

Powermonger jest lub będzie prawdopodobnie jedną z najlepszych gier tego roku. Program został napisany przez grupę BULLFROG czyli przez ludzi „odpowiedzialnych" za tak znakomite programy jak POPULOUS 1 i 2. Jest to kolejna gra, którą można zakwalifikować gdzieś pomiędzy role-playing i strategy.

Jedyną, ale dość istotną różnicą jest fakt, że w POPULOUS trzeba było budować jak największe domy i „bić" wroga. POWERMONGER wymaga, abyś był jak najbardziej chciwy i nieuczciwy w stosunku do Twojego przeciwnika.

Twoje zadanie polega na zdobywaniu zamieszkałych (lub nie) wysp. Liczba opcji dostępnych w tej grze jest dość duża, jednakże wszystkie są rozmieszczone tak jak w POPULOUS (czyli na dole ekranu). Z lewej strony ekranu jest wyświetlana mapa informująca Cię gdzie się aktualnie znajdujesz lub gdzie należałoby szukać przeciwnika.

Oryginalna wersja gry zawiera dwie dyskietki. Dzięki temu grę można rozpocząć albo od czasów prehistorycznych lub też od czasów drugiej wojny światowej. Samo miejsce nie zmienia jednak zasad

gry — dalej chodzi o zdobywanie terenów i pobicie wroga.

Do niejakiej konfuzji może jednak doprowadzić chyba nieco zbyt bogate menu. Stosunkowo wiele znaków nie kojarzyło mi się zupełnie z niczym i gdybym nie miał instrukcji obsługi z grania byłyby nici.

Choć nie jestem miłośnikiem programów tego typu, to jednak poświęciłem parę dni wakacji na rozpracowanie tej gry. Dobra grafika, niezłe efekty dźwiękowe i stosunkowo ciekawa fabuła pozwalają mi ją polecić wszystkim fanom komputerowej iluzji.

OLAF PRZYBYSZEWSKI

HARPOON BATTLESET 4

Dzięki firmie IPS dotarł do Polski czwarty zestaw scenariuszy do symulacji powietrzno-morskiej — Harpoon. Areną działań jest tym razem Zatoka Perska i Ocean Indyjski. Obszar ten jest gorący nie tylko ze względu na położenie geograficzne, ale i stosunki międzynarodowe, napięcia społeczne oraz antagonizmy religijne. Państwa tego regionu znane są z wybujałych ambicji i niestałości sojuszy politycznych i militarnych. Mają jednocześnie dość pieniędzy na zakup nowoczesnych systemów uzbrojenia. Zagaszone z wielkim trudem przez społeczność międzynarodową lokalne konflikty mogą wybuchnąć na nowo z bardzo błahego powodu ogarniając ogniem wojny cały region.

Zachowanie równowagi militarnej i politycznej w tym rejonie jest sprawą bardzo ważną z uwagi na duże zasoby ropy naftowej tak niezbędnej do prawidłowego funkcjonowania przemysłu na całym świecie. Całkowite opanowanie tego regionu przez jeden kraj mogło by spowodować wywieranie presji na kraje „wolnego świata" poprzez groźbę wstrzymania dostaw ropy. Unormowanie sytuacji jest tym trudniejsze iż kraje te „lawirują" w swoich sympatiach między Wschodem i Zachodem chcąc czerpać korzyści w postaci broni i pieniędzy z obu stron. Jedynym wyjątkiem jest tu Iran pałający nienawiścią do wszystkiego co nie islamskie, a zwłaszcza do „imperialistycznego" Zachodu.

Tak oto przedstawia się tło wydarzeń szesnastu scenariuszy ułożonych dla nas przez programistów Electronic Arts. Jak zwykle na początku gry musimy wybrać stronę konfliktu, którą będziemy dowodzić; tym razem mamy do wyboru „Blue" („niebieskich") i Red („czerwonych"). Bliższe przyjrzenie się scenariuszom wyjaśnia tę zagadkę": stronami w konfliktach są kraje Środkowego Wschodu, a USA i ZSRR (a może Wspólnota Niepodległych Państw) wkraczają z chwilą gdy szala zwycięstwa przechyla się na niekorzyść sojusznika. Określenie kto jest „Czerwony", a kto „Niebieski" wynika z zasady: kraje pragnące zmienić układ sił w regionie na drodze militarnej są „czerwone", a siły walczące o zachowanie równowagi — „niebieskie".

Pierwsze dwa scenariusze są wprawką do poważniejszych zadań — rozgrywają się podczas wojny iracko-irańskiej. Natomiast trzeci jest przykładem jakie są maksymalne możliwości gry Harpoon. Rozgrywa się on w bardzo interesujących warunkach — jest to początek operacji Pustynna Burza (!). W akcji tej po obu stronach bierze udział ponad tysiąc dwieście samolotów. Nowością w tym scenariuszu jest użycie „niewidzialnego samolotu" F-117A Stealth, bombowców B-52 oraz rakiet Tomahawk o zasięgu ponad tysiąca mil. Z instrukcji wynika, że mogą wystąpić problemy z brakiem pamięci podczas rozgrywania tego scenariusza, ale oprócz znacznego zwolnienia czasu podczas gry, nie zauważyłem żadnych niepokojących objawów przy 1 MB RAM zainstalowanej w moim komputerze.

Kolejne scenariusze to już swobodna i niczym nieskrępowana twórczość programistów. Tłem dla nich są rozważania co by było gdyby Amerykanie nie podjęli natychmiastowych działań po inwazji Kuwejtu i zostawili Arabię Saudyjską na łasce Iraku; o ile trudniej i jakim kosztem odbyłoby się późniejsze zdobycie przewagi powietrznej w Zatoce. Scenariusze dotyczące Zatoki Perskiej wieńczy akcja antyterrorystyczna przeciwko Iranowi.

Następne operacje rozgrywają się na Oceanie Indyjskim. Ocean ten jest doskonałym miejscem na ukrycie zespołu lotniskowców nie mówiąc już o łodziach podwodnych. Zwłaszcza tych nowych mogących operować na bardzo dużych głębokościach. Być może w lokalizowaniu nieprzyjaciela pomoże kolejna nowinka — satelity szpiegowskie. Nowy obszar działania to nowe kłopoty. Indie, dynamicznie rozwijający się kraj tego regionu posiadający w miarę nowoczesną marynarkę i lotnictwo dąży do przejęcia kontroli nad Zatoką Bengalską, potencjalnym źródłem ropy naftowej. Nie mogą na to pozwolić inne kraje tego regionu: Bangladesz, Tajlandia, Indonezja i Sri Lanka. Mają jednak za małe siły na powstrzymanie agresywnych zamierzeń Indii, proszą więc o pomoc jej odwiecznego wroga Pakistan. Zaniepokojone tym faktem Indie ubiegają się o wsparcie ZSRR. Na co Pakistan reaguje żądaniem pomocy od USA. Tak oto konflikt lokalny przeradza się w globalny.

Według tych wydarzeń mamy do rozegnania dwa scenariusze pod wspólnym tytułem Wojna Bengalska. Wątkiem wiążącym kolejne akcje jest zapewnienie dopływu ropy dla gospodarki zachodniej. Może to być nieco utrudnione z uwagi na fakt, że ZSRR właśnie zaatakował Iran, a Syria i Irak — Turcję. Mamy w tych scenariuszach do czynienia z szerokimi działaniami powietrzno-morskimi. Biorące w nich udział jednostki „będą" najnowocześniejszymi na świecie. Będą — bowiem programiści przewidując, że akcje te rozegrają się w przyszłości, wprowadzili nową klasę łodzi podwodnych i lotniskowca. Dodatkowo aby utrzymać siłę ognia amerykańskich jednostek, mniej licznych po cięciach budżetowych w Kongresie, wprowadzono na uzbrojenie nowe i niesamowicie efektywne rakiety o nazwie Deadeye.

Cechą charakterystyczną czwartego zestawu scenariuszy są różnego rodzaju niespodzianki. Już od początku mamy do czynienia z mieszaniną sprzętu i uzbrojenia. Radzieckie Migi 19 i 21 produkcji chińskiej razem ze starszymi wersjami F-4E Phantom i F-105 Tiger. Angielskie i francuskie niszczyciele zdemobilizowane po drugiej wojnie światowej uzbrojone w pociski Harpoon. Efektywne dowodzenie takim zespołem, który posiada długodystansowe rakiety ale bardzo słabe radary wymaga nie lada umiejętności. Pikanterii rozgrywce dodaje pojawienie się w scenariuszach trzeciej strony. Należy bardzo dokładnie zidentyfikować cel, zwłaszcza jeżeli jest nim łódź podwodna aby nie posłać na dno sprzymierzeńca co mogło by wywołać bardzo poważne reperkusje polityczne...

Z czystym sumieniem mogę polecić czwarty zestaw scenariuszy fanom gier strategicznych. Odniesienie zwycięstwa wymaga sporo czasu oraz treningu i czasami może przyprawić o ból głowy, ponieważ warunkiem sukcesu jest nie tylko zadanie przeciwnikowi znacznych strat, ale i wykonanie zadania z minimalnymi stratami własnymi.

W opisie do niektórych scenariuszy jest notatka, aby przeczytać plik HELP.TXT (zapisany na dyskietce) w przypadku kłopotów z pamięcią. Problem polega na tym, że takiego pliku w wersji dla Amigi po prostu ... nie ma. Podejrzewam, że ta notatka dotyczy wersji programu dla komputerów PC.

PAWEŁ GALAS

DYSTRYBUTOR:
IPS Computer Group, Warszawa, ul. Okrężna 3, tel. 642-27-66
Program: **Battle Set No 4 (IOPG)**
Rodzaj gry: strategiczna
Producent: **Electronic Artc**
Komputer: **Amiga, IBM**
Wymagania: Działa tylko wraz z grą Harpoon (dla IBM wymagana jest wersja 1.2.1 lub wyższa)

KLAX

Jestem wielbicielem gier logicznych, tak więc, gdy zobaczyłem tą grę, wpadłem w iście cielęcy zachwyt. KLAX jest kolejną mutacją przesławnego TETRISA.

Łapiesz spadające, różnokolorowe cegiełki, na wózek, i następnie opuszczasz je na półkę w pięciu różnych przedziałach. Kiedy spotkają się trzy cegiełki w tym samym kolorze, znikają i zostaje Ci doliczona pewna ilość punktów. W miarę upływu czasu cegiełki zaczynają spadać coraz prędzej, co utrudnia ich łapanie i upuszczanie w odpowiednie miejsca. Jednorazowo na wózek można zabrać 5 cegiełek i potem, jeżeli zwolni się miejsce na półce dla nowego koloru, upuścić cegiełkę.

Od czasu do czasu pojawia się element mieniący się różnymi kolorami. Można go upuścić na dowolny stosik, a przyjmie on jego kolor. Czasami też może się przywlec niechciana cegiełka. Można ją wtedy na chwilę podrzucić, aby zabrać tę pożądaną. „Klaxy", czyli grupy minimum trzech cegiełek, można układać poziomo (leżące obok siebie elementy w tym samym kolorze), pionowo lub ukośnie. Im dłuższa grupa tym więcej punktów...

Ciekawe, czy komus uda się przejść po kolei 6 poziomów...

PIOTR LISZEWSKI

DYSTRYBUTOR:
IPS Computer Group, ul. Okrężna 3, 02-916 Warszawa, tel. 642-27-66, 642-27-68.

- Tytuł : **KLAX**
- Producent : **DOMARK**
- Komputer : **Commodore 64**
- Nośnik : Kaseta i dyskietka
- Cena : 39 000 zł
- Wymagania : kolorowy telewizor lub monitor



BIG BOX

Wiemy wiemy, mało u nas opisów gier dla C-64 za co Czytelników przepraszamy, choć to niezupełnie nasza wina. Firmy sprowadzające programy do Polski zapominają o nieszczęsnym Commodorku lub twierdzą, że jest to nieopłacalne. Swoje twierdzenia popierają faktem, że czasy „małego Atari", „Spektrumny" i pozostałych ośmiobitowców dawno już przeminęły, tak więc i C-64 powinien upomnieć się o emeryturę. Dziwne jest tylko to, że w krajach wyżej rozwiniętych pod kątem komputerowym, małe Commodorki nadal są używane w najprzeróżniejszych sytuacjach a tamtejsze firmy oferujące software mogą pochwalić się dużym wyborem NAJNOWSZYCH (!) gier dla C-64.

IPS jest jedną z niewielu (o ile nie jedyną) firmą oferujących gry komputerowe. W jej ofercie obok gier dla Amigi, PC i ST znalazło się także kilka tytułów z wdzięczną nalepką „C-64". Do nich między innymi należy duży zestaw gier zatytułowany BIG BOX — zawiera on aż 30 gier na 6 kasetach. A oto tytuły gier:
ACE 2088, ALIENS, BIG TROUBLE IN LITTLE CHINA, CHAMPIONSHIP BASEBALL, CHAMPIONSHIP BASKETBALL, CHAMPIONSHIP FOOTBALL, CORPORATION, DANDY, ENDURO RACER, EXPLORER, FIRE TRAP, GEE BEE AIR RALLY, GALACTIC GAMES, GHOSTBUSTERS, GUADALCANAL, HACKER, HIGH FRONTIER, INCREDIBLE SHRINKING SPHERE, KARNOV, KNIGHTMARE, MERMAID MADNESS, PARK PATROL, PRODIGY, RAMPAGE, REAL GHOSTBUSTERS, SPINDIZZY, STAR RAIDERS 2, SUPER HANG ON, SUPERSPRINT, WONDERBOY.

Wraz z kasetami w pudełku znajduje się instrukcja i jednocześnie opis wszystkich gier, niestety tylko w języku angielskim.

Aby opisac wszystkie te gry trzeba by przeznaczyć na ten cel sporą część C&A; ponieważ jest to niemożliwe, z konieczności ograniczę się do krótkich charakterystyk wybranych gier, które najbardziej mi się podobały.

WONDER BOY (ACTIVISION)

Gra zręcznosciowa typu „aby do przodu" spotykana na automatach. Przygody małego chłopca na tajemniczej wyspie. Zadaniem bohatera jest uwolnienie dziewczynki z łap potworów. Przystępna kolorowa grafika i dźwięk.

GUADALCANAL (ACTIVISION)

Strategiczna. Bitwa morska w zatoce. Grafika raczej symboliczna podobnie jak i dźwięk.

ALIENS (ELECTRO DREAMS)

Gra labiryntowa, na podstawie filmu „Obcy — decydujące starcie". Prowadzisz ekipę złożoną z wytrawnych komandosów, po bazie opanowanej przez potwory z kosmosu. Pole akcji oglądasz oczami aktualnego bohatera. Ładna grafika, w przeciwieństwie do dźwięku, uzupełnianego zaskakującą akcją.

SPINDIZZY (ELECTRO DREAMS)

Znowu „labiryntówka", lecz tym razem trójwymiarowa. Poruszasz się bączkiem po najeżonym pułapkami labiryncie w poszukiwaniu diamentów. Tak jak poprzednio, brak muzyki uzupełnia bardzo dobra grafika.

RAMPAGE (ACTIVISION)

Zamiast znęcać się nad sąsiadką grając w „kosza" kulą armatnią, rozwal sobie kilka domów wcielając się jedno z trzech monstrów. Gra jak się domyślasz zręcznościowa, z dużym poczuciem humoru. Grafika i muzyka w normie.

BASKETBALL (GAMESTAR)

Firma ta znana jest z produkcji doskonałej jakości gier sportowych. Koszykówka pozwala poczuć się uczestnikiem NBA nawet tym, którzy na co dzień mają do czynienia wyłącznie z koszem na śmieci.

KNIGHTMARE (ACTIVISION)

Jest to jedna z pierwszych gier typu role-playing. Jako wygnaniec (zaczynasz w więzieniu) toczysz walkę z siłami zła. Rewelacyjna grafika i animacja, z najdrobniejszymi szczegółami. To jest to, „co tygrysy lubią najbardziej"!

GEE BEE AIR RALLY (ACTIVISION)

Wyścigi, nie na ziemi, ani na wodzie lecz w powietrzu przy użyciu małych, jednosilnikowych samolotów. Rarytas dla lubiących prędkość i lotnictwo.

MERMAID MADNESS (ELECTRO DREAMS)

Zdesperowany „zagłaskiwaniem na śmierć" przez syrenę nurek wskakuje do wody w poszukiwaniu spokoju. Niestety, stary wrak okrętu łapie go w swoją pułapkę. Ty jako owa nieszczęsna syrenka (nieco ostatnio przytyła), rzucasz się na pomoc uwięzionemu nurkowi. Musisz go uratować nim skończy się mu powietrze. Bardzo ciekawa gra obfitująca w przeładowane humorem sytuacje.

PARK PATROL (FIREBIRD)

Sam nie wiem czemu lubię tę grę. Jako strażnik (lub strażniczka) parku, zajmujesz się zbieraniem puszek i butelek po Coli, wyławianiem topielców i tępieniem szkodników. Przeszkadzają ci w tym hordy węgorzy, złośliwe kraby i równie złośliwi wypoczynkowicze. Grafika i muzyka, jak na czasy powstawania tej gry, bardzo ładne.

KARNOV (DATA EAST)

Przygody faceta spod znaku młota i sierpa w labiryncie. Grafika i muzyka w normie.

To tylko jedenaście spośród 30 naprawdę niezłych gier. Uważam, że tym razem firma IPS postarała się naprawdę przypominając sobie użytkownikach Commodore 64 i magnetofonu. Programy są nagrane porządnie i na firmowym magnetofonie na którym nie przeprowadzano prób rozregulowania głowicy nie miałem żadnych problemów z ich wczytaniem.

O ile mi wiadomo to nie koniec rozrywki dla właścicieli C-64. W chwili gdy to piszę koledzy kończą już pisać recenzje kilku innych programów — i do nich zapewne powrócimy za miesiąc.

PIOTR LISZEWSKI

DYSTRYBUTOR:
IPS Computer Group, ul. Okrężna 3, 02-916 Warszawa, tel. 642-27-66, 642-27-68.

- Tytuł zestawu: **BIG BOX**
- Producent: **BEAU-JOLLY**
- Komputer: **Commodore 64/128 w trybie C-64**
- Nośnik: **kaseta**
- Cena: **100 000 zł**

LICENCE TO KILL

Czy pamiętacie Seana Connery albo Rogera Moora w roli Jamesa Bonda? Teraz ich rolę jak pewno wiecie gra Timothy Dalton. Z komputerami ma to tyle wspólnego, że po prostu przeniesiono przygody znanego superagenta z ekranu filmowego na komputerowy. Nie jest to zresztą pierwszy taki program — wystarczy przypomnieć tu choćby grę RAMBO czy symulator FLIGHT OF THE INTRUDER.

„LICENCE TO KILL" nie jest grą rewelacyjną ani najnowszą, lecz przyciągnie zapewne graczy zakochanych we wszelakiego rodzaju strzelaninach (jako, że fabułą tej gry jest ogień ciągły). Strzelanina ta polega na locie helikopterem i strzelaniu do wszystkiego co się rusza i usiłuje strzelić do Ciebie. Musisz niestety pamiętać o jednym: gdy lecisz helikopterem, możesz nim się poruszać w różne strony, nie masz jednak możliwości regulacji szybkości lotu. Gdy chcesz zwolnić, Twój śmigłowiec obniża lot. Jaki jest z tego morał? Że każdy udany lot charakteryzuje się taką samą liczbą startów i lądowań...

Po prawej stronie masz pole informujące o stanie technicznym Twojego wehikułu. Jeśli helikopter lub jego dowolna część „mruga", to znaczy, że przeciwnik odgryzał się skutecznie i być może za chwilę zamienisz się wraz ze śmigłowcem w kulę ognia.

Lot helikopterem zajmuje cały poziom pierwszy. Po przejściu do poziomu drugiego stajesz na własnych nogach i ziejąc ogniem ciągłym nadal kładziesz pokotem nieprzeliczone zastępy nieprzyjaciela zesłane na Ciebie prosto z wyobraźni programistów. Co się stanie na poziomie trzecim pozostawiam już ciekawości graczy.

Do niewątpliwych zalet tego programu należy fakt, że możesz go kupić jako LEGALNY za 39000 (słownie: trzydzieści dziewięć tysięcy) złotych. Na pewno nie można powiedzieć, że jest to cena wygórowana, choć jak znam życie malkontentów nigdy nie zabraknie.

OLAF PRZYBYSZEWSKI

DYSTRYBUTOR:
IPS Computer Group, ul. Okrężna 3, 02-916 Warszawa, tel. 642-27-66, 642-27-68.

- Tytuł: **LICENCE TO KILL**
- Producent:
- Komputer: **Amiga, Commodore 64** (kaseta/dyskietka)
- Cena: 39 000 złotych
- Wymagania: —

NIEBO NAD GŁOWĄ I W... KOMPUTERZE

Czy zastanawiałeś się, patrząc nocą na pogodne, rozgwieżdżone niebo, jak też ono wygląda nad Australią lub nad równikiem? Albo jakie niebo oglądali starożytni budowniczowie piramid, czy też... trzej królowie, wędrujący do Betlejem? Czy zdarzyło Ci się oglądać wschód Słońca i świecącą nisko nad horyzontem Wenus? Niektóre z tych zjawisk jeszcze na pewno zobaczysz, ale musiałbyś dożyć matuzalemowego wieku, aby ponownie zobaczyć kometę Halleya, której prawdopodobnie nie udało Ci się zobaczyć w roku...? chociaż bardzo chciałeś.

No właśnie, przypuszczam, że jesteś amatorem astronomem a może tylko interesujesz się trochę astronomią? Jeżeli tak, to na pewno zechcesz zobaczyć to, o czym wspomniałem i wiele wiele innych bardzo ciekawych zjawisk. Zobaczyć już, zaraz, natychmiast na... ekranie Twego monitora lub telewizora.

Ale po kolei. Taką arcyciekawą podróż pozwala Ci odbyć mało znany program o nazwie SKY TRAVEL opracowany przez firmę Deltorn w 1984 roku i jest to jeden z programów w którym data produkcji zupełnie się nie liczy.

SKY TRAVEL to w założeniu domowe planetarium czyli tzw. program edukacyjny, ale tak naprawdę jest to magiczny wehikuł, którym odbędziesz wspaniałą darmową podróż w czasie i przestrzeni.

Program ten jest, jak piszą jego twórcy, prawdziwym oknem na galaktykę, otwieranym w dowolnej chwili z dowolnego miejsca na Ziemi. Okno to ma kąt widzenia 72 stopnie czyli taki jak oko ludzkie, ale można ten kąt zmieniać aż do 90 stopni, co sprawia, jakbyś oglądał niebo przez niewielką lunetkę. Wtedy możesz zobaczyć więcej niż gołym okiem.

Po uruchomieniu programu zobaczysz wygląd nocnego nieba oglądanego w dniu 1 stycznia 1985 nad Waszyngtonem. Popatrz na ekran: u góry po prawej stronie widnieje data JAN 01 1985 (1 styczeń 1985) a poniżej — współrzędne geograficzne miejsca, z którego jest oglądane niebo. Nieco niżej zobaczysz wyświetlony w rewersie napis SKY. Jest to jedna z czterech opcji programu, wywoływanych klawiszem F1. Aby przeprowadzić się z Waszyngtonu do miejsca gdzie mieszkasz wywołaj opcję MAP (naciskając F1 kilkakrotnie) i wciśnij RETURN. Z dyskietki zostanie wczytana i pojawi na ekranie mapa świata w tzw. projekcji Mercatora. Zobaczysz też kursor (jasny krzyżyk) w położeniu odpowiadającym lokalizacji Waszyngtonu. Za pomocą joysticka przesuń kursor do Polski. Jeżeli znasz położenie geograficzne miejscowości w której mieszkasz możesz ustawić kursor dokładnie nad nią, kontrolując liczby pojawiające się w okienku informacyjnym przy słowie LATITUDE (szerokość geograficzna) i LONGITUDE (długość geograficzna). Przy okazji dowiesz się być może po raz pierwszy, jaki jest Twój ":adres" geograficzny. Podpowiem Ci: jeśli mieszkasz w Warszawie, to Twój adres jest: szerokość 52 stopnie i 13 minut N, długość zaś wynosi 21 stopni i 7 minut E. Jeśli mieszkasz w innej części Polski sięgnij po prostu do atlasu...

Teraz kolej na ustawianie zegara na dzisiejszą datę. Wciskaj klawisz F1 aż zobaczysz u dołu po prawej stronie napis SET. Wtedy, manipulując joystickiem, wybierz kolejno miesiąc, dzień oraz godzinę obserwacji. Na końcu tego wiersza z „zegarem cyfrowym" zobaczysz literkę A lub P; oznacza to, że pora jest przed południem (A) lub po (P). Nastawiając zegar na przykład na 15 wrzesień 1992 godzinę 2320 musisz ustawić:

SEP 15 1992 AD, 11:20:OP.

Wracamy teraz do mapy nieba (wciskaj F1 aż pojawi się SKY, potem wciśnij RETURN). Program informuje, w wierszu u dołu ekranu, że oblicza położenia ciał niebieskich i po chwili widzisz obraz taki, jaki widziałbyś stojąc twarzą zwróconą na południe i patrząc na rzeczywiste niebo. Spróbuj teraz „obracsć się" w prawo, tj. ku zachodowi, naciskając joystick w prawo. Po chwili „zastanawiania się" komputer narysuje obraz nieba najpierw w kierunku południowo-zachodnim, następnie zaś — zachodnim. Zwróć uwagę na wygląd nieba: niektóre gwiazdy połączone są liniami, tworząc w ten sposób gwiazdozbiory, których nazwy, w postaci trzyliterowych skrotów są Ci prawdopodobnie znane. Zwróć uwagę, że w pobliżu gwiazdozbioru Skorpiona (Sco) widnieją dwa dziwne znaki. Są to symbole znajdujących się w tym miejscu planet Urana i Saturna.

Wciśnij klawisz "+" i spójrz na okienko informacyjne. Zobaczysz, że cyfra przy słowie RATE zmieniła się: zamiast 0X jest teraz 1X. Wciśnij "+" kilkakrotnie: pojawi się 2X, 4X..., 32X i 64X. Niebo drgnęło. Gwiazdy i gwiazdozbiory zaczynają się poruszać jednocześnie, zegar ruszył i idzie coraz szybciej, jest już godzina 11:00P, 11:30P, zbliża się północ... Zatrzymaj czas, wciskając kilkakrotnie "-" tak, by RATE powróciło do 0X. Zatrzymałeś obrót sfery niebieskiej. Wykorzystaj przerwę i przywołaj do swego planetarium inne ciała niebieskie. Najpierw Księżyc i Słońce: wciśnij klawisz F5 (FIND). Gdy zobaczysz MOON — wciśnij RETURN.

Twórcy programu napisali na jego temat sporą, 120. stronicową książeczkę, nie sposób więc opisac wszystko w krótkim artykule. Więc tylko najważniejsze informacje:

SKY TRAVEL dysponuje 4 trybami: MAP, SET, i CHART, które są wywoływane klawiszem F1 i akceptowane przez RETURN. W trybie SKY pojawia się obraz nieba i okienko informacyjne które pozostaje w innych trybach. W trybie SKY za pomocą klawisza F3 można wywoływać różne dodatkowe opcje jak np. LINES (linie łączące gwiazdozbiory), SYMBOLS (symbole planet), NAMES (nazwy gwiazdozbiorów) i inne. Możesz też zmieniać kąt widzenia (klawisz SHIFT i "+" lub "-" od 72 stopni do 9 stopni, wprowadzając do oglądania nieba „teleskop" o niedużym powiększeniu. Natomiast opcją DEEP przywołasz na ekran około 100 najbardziej interesujących obiektów pozagalaktycznych.

W trybie CHART na ekranie pojawi się widok nieba na białym tle z równoleżnikami i południkami niebieskimi; jeżeli masz drukarkę, możesz ten obraz wydrukować (klawisze SHIFT i P). Tryb FIND służy do lokalizowania na środku ekranu niektórych ciał niebieskich. Jeżeli chcesz dowiedzieć się czegoś więcej o wyszukanym przez FIND obiekcie, wciśnij F7 (INFORM).

SKY TRAVEL potrafi wiele: pokaże Ci wygląd nieba sprzed 10000 lat i za 10000 lat, lub w dowolnej chwili w tym przedziale czasu, umie odtworzyć przebieg zaćmienia Słońca, Księżyca lub planet oraz tzw. przejścia planet przez tarczę Słońca, (pokaże Ci przemieszczanie się komety Halleya w czasie jej ostatniego zbliżenia do Ziemi w latach 1985—86 a nawet umie pokazać prawdziwą gwiazdę betlejemską).

Na zakończenie proponuję, abyś wybrał się w daleką podróż w celu obejrzenia komety Halleya, gdyż w Polsce tylko niewielu szczęśliwców mogło ją oglądać. Podróżujesz do Australii, wiosną 1986 roku: klawiszem F1 wywołaj MAP, wybierz LATITUDE — 24 stopni S, LONGITUDE 137 stopni E, wywołaj SET, ustaw zegar na 7 kwietnia (APR) 1986 AD, godz. 11:30 PM. Wywołaj SKY potem klawiszem F5 wywołaj KOMET, zatwierdź przez RETURN. Oto na ekranie południowe niebo nad Australią i kometa Halleya w całej swej krasie przed Twoimi oczami.

DD





TELETEKST DLA COMMODORE 64

Co to jest teletekst?

Teletekst jest to cyfrowy system przesyłania dodatkowych informacji jednocześnie z programem telewizyjnym. Informacje te można oglądać na ekranie telewizora (w naszym przypadku na ekranie monitora komputerowego lub telewizora służącego jako minitor). Telegazeta zawiera szereg wiadomości z kraju i ze świata, aktualności sportowe, kursy walut, program TV, auto giełdę, reklamy — jednym słowem jest to bank danych z prawdziwego zdarzenia.

Sygnałem teletekstu mogą być programy 1 i 2 Telewizji Polskiej i Czechosłowackiej. Odbiór sygnału jest możliwy przy pomocy odbiornika telewizyjnego lub magnetowidu. Sygnałem teletekstu mogą być także programy telewizji satelitarnej. Odbiór w tym wypadku jest możliwy przy pomocy tunera telewizji satelitarnej wyposażonego w wyjście sygnału wizyjnego na złączu typu CINCH lub SCART. Maksymalny czas na oczekiwanie (pojawienie się) strony wynosi 24 sekundy.



KONSTRUKCJA I GWARANCJA

Zestaw do odbioru teletekstu składa się z:

1. Procesora teletekstu — jest to „pudełeczko" do którego doprowadza się sygnał niskiej częstotliwości (z magnetowidu wyposażonego w tuner TV lub telwizora).
2. Przewodu wizji (2 * CINCH) — to właśnie tym kablem doprowadza się sygnał niskiej częstotliwości.
3. Specjalnego przewodu połączeniowego niezbędnego do połączenia procesora teletekstu z komputerem.
4. Modułu z programem obsługującym teletekst.
5. Instrukcji obsługi z kartą gwarancyjną. Producent udziela 12 miesięcy gwarancji na swój wyrób na standardowych warunkach.

INSTRUKCJA OBSŁUGI

Instrukcja obsługi napisana jest w języku polskim. Nie jest ona niestety najpiekniej wydana, lecz w przystępny i zrozumiały sposób opisuje zasady korzystania z zestawu. Brakowało mi w niej tzw. ściągawki czyli opisów funkcji wszystkich klawiszy zebranych na jednej stronie. Sam program jest wyposażony w ekrany pomocnicze. W czasie działania programu użytkownik może skorzystać z kilku ekranów pomocniczych przywoływanych w dowolnej chwili i zawierających krótki opis poszczególnych funkcji.

INSTALACJA

Proces instalacji nie jest niestety zbyt prosty. W celu zabezpieczenia komputera przed uszkodzeniem konieczne jest zachowanie kolejności czynności podczas instalacji i demontażu zestawu.

Aby zmontować zestaw należy:

— wyłączyć komputer
— połączyć procesor z komputerem za pomocą specjalnego przewodu
— przyłączyć procesor teletekstu do telewizora, magnetowidu lub tunera satelitarnego (mówiąc krótko: do sygnału teletekstu o niskiej częstotliwości)
— zainstalować moduł w porcie użytkownika (USER PORT)
— włączyć komputer.

MOŻLIWOŚCI ZESTAWU

Testowany przeze mnie zestaw umożliwia:

— wywołanie dowolnej strony teletekstu
— wyszukanie i przygotowanie drugiej strony podczas przeglądania pierwszej
— zaprogramowanie wybranych numerów stron pod klawiszami funkcyjnymi
— zaprogramowanie listy kolejno wywoływanych numerów stron
— zapis strony do pamięci jednego z 48 banków stron
— zmianę atrybutów wyświetlanych stron, możliwość wyświetlenia tylko grafiki lub tylko tekstu
— zapis i odczyt wybranej strony na dyskietce lub kasecie
— wywołanie uproszczonej listy komend (tzw. HELP)

W AKCJI

Podczas testu źródłem sygnału teletekstu był 2 program TVP. Sygnał o niskiej częstotliwości pobierałem z magnetowidów JVC HRR-D210 i PANASONIC NV-J35.

To, co wyróżnia komputerową telegazetę od normalnych telegazet montowanych w telewizorach, to przede wszystkim możliwość wydrukowania strony na drukarce czy zapisania jej na dowolnym nośniku magnetycznym, wyszukiwanie i przygotowanie drugiej strony podczas przeglądania aktualnej, przypisanie wybranych numerów stron określonym klawiszom funkcyjnym. Oprócz tego możliwe jest również zaprogramowanie listy kolejno wywoływanych stron, zapis strony teletekstu do pamięci jednego z 48 banków stron i zmiana atrybutów wyświetlanych stron. Być może w najnowszych dekoderach teletekstu niektóre z powyższych możliwości są już dostępne.

Niestety zalety nie są wyłączną domeną takich przystawek, choć uwaga ta odnosi się do wszystkich urządzeń tego typu, nie tylko do zestawu tu opisywanego. Minusami wersji komputerowych są przede wszystkim:

— brak wygodnej obsługi za pomocą pilota,
— brak możliwości tworzenia obrazu MIX tzn. obrazu telegazety nałożonego na obraz telewizyjny,
— brak możliwości podsłuchu dźwięku TV podczas czytania telegazety.

Testowany zestaw umożliwiał ponadto:

— wyświetlenie na ekranie tylko grafiki,
— wyświetlenie na ekranie tylko tekstu,
— wyświetlenie na ekranie tylko grafiki i atrybutów związanych z grafiką,
— to co powyżej tylko z tekstem,
— wyświetlenie na ekranie grafiki oraz tekstu znakami graficznymi,
— to co powyżej tylko z tekstem,
— zwiększanie, zmniejszanie numerów wyświetlanych kolorów o jeden,
— zmianę kolorów znaków tekstowych lub graficznych.

NASZ WERDYKT

Opisany powyżej zestaw w pełni realizuje postawione przed nim funkcje. Dużą zaletą jest możliwość uzyskania polskich znaków (od pierwszego lipca cała telegazeta będzie nadawana z polskimi znakami) i prostota obsługi. Przydało by się tylko, żeby Polska Telegazeta była bardziej rozbudowana, ale to jest temat na zupełnie inny artykuł.

MARIUSZ FERDYN

PRODUCENT: MIAN ELECTRONICS S.C.
53-307 WROCLAW, ul.Kamienna 23/39
tel. (071) 675733
CENA: 880.000 zl

ZALETY:

— umieszczenie programu obsługującego na module
— współpraca z magnetofonem i stacją dysków
— łatwa obsługa
— możliwość uzyskania międzynarodowych znaków w tym polskich.

WADY:

— krótkie kable połączeniowe
— możliwość włożenia modułu do „góry nogami" co prawdopodobnie nie może spowodować uszkodzenie komputera.



Odcinek dla poczty

Zł ....
Słownie zł ....
....
Wpłacający ....
....
Dokładny ....
adres ....
i kod ....

Wydawnictwo BAJTEK
Warszawa, ul. Wspólna 61

Bank Agrobank S.A. 470005-1834-131
ul. Grochowska 262
04-398 Warszawa

Opłata

Datownik

podpis przyjmującego

Odcinek dla posiadacza rachunku

Zł ....
Słownie zł ....
....
Wpłacający ....
....
Dokładny ....
adres ....
i kod ....

Wydawnictwo BAJTEK
Warszawa, ul. Wspólna 61

Bank Agrobank S.A. 470005-1834-131
ul. Grochowska 262
04-398 Warszawa

Opłata

Datownik

podpis przyjmującego

Potwierdzenie dla wpłacającego

Zł ....
Słownie zł ....
....
Wpłacający ....
....
Dokładny ....
adres ....
i kod ....

Wydawnictwo BAJTEK
Warszawa, ul. Wspólna 61

Bank Agrobank S.A. 470005-1834-131
ul. Grochowska 262
04-398 Warszawa

Opłata

Datownik

podpis przyjmującego

Odcinek do wysłania

Zł ....
Słownie zł ....
....
Wpłacający ....
....
Dokładny ....
adres ....
i kod ....

Wydawnictwo BAJTEK
Warszawa, ul. Wspólna 61

Bank Agrobank S.A. 470005-1834-131
ul. Grochowska 262
04-398 Warszawa

odpis

Datownik

podpis przyjmującego

Tytuł \ Liczba kolejnych zeszytów	3	6	12	po ile egz.
Bajtek	X	60000	120000	
C&A	30000	60000	X	
TOP SECRET	27000	54000	X	

Wpłat dokonywać na konto:
Spółdzielnia "BAJTEK"
Bank "Agrobank S.A."
470005-1834-131
ul. Grochowska 262
04-398 Warszawa

tu zanotuj, co zamówiłeś

Warunki prenumeraty:

- Prenumerata zawarta przed upływem ważności kuponu gwarantuje niezmienność cen
- Przesyłka pocztowa nie wymaga dodatkowych opłat
- Jeżeli w ciągu 2 tyg. od pojawienia się numeru w kioskach przesyłka nie nadeszła, prosimy o kontakt
- Za błędy wynikające z niestarannnego wypełnienia formularza redakcja nie ponosi odpowiedzialności
- Prosimy o staranne i wyraźne zakreślenie odpowiednich ilości egzemplarzy



Szanowni Czytelnicy!

Jak zapewne sobie przypominacie, w tym numerze C&A powinny być zamieszczone wyniki ankiety, jaką przeprowadziliśmy w C&A nr 4/92. Wasz odzew przeszedł jednak nasze najśmielsze oczekiwania, redakcja została wręcz zasypana wypełnionymi ankietami. W związku z tym potrzebowaliśmy nieco więcej czasu, aby tę masę danych przerobić na przystępną dla Czytelników formę i opublikować. Dlatego informujemy, że wyniki ankiet wraz z listą nagrodzonych zaprezentujemy Wam już na pewno w numerze październikowym. Za poślizg przepraszamy i dziękujemy za tak duże zainteresowanie.

Jeśli zaś chodzi o konkurs RELAX: nadesłane prace (a przysłaliście ich grubo ponad sto) są już od pewnego czasu oceniane i ich omówienia należy się spodziewać w listopadzie. Powiem tu jeszcze tylko, że jest sporo naprawdę świetnych programów i czasem trudno się zdecydować, któremu przyznać palmę pierwszeństwa.

I jeszcze z innej beczki: jak do tej pory nikomu nie daliśmy pozwolenia na dystrybucję naszych dyskietek, zarówno amigowskich, jak i dla C-64. To samo dotyczy kaset. Jeżeli więc ktoś zaoferuje Wam programy publikowane w C&A, możecie być pewni, iż pochodzą one z nielegalnego źródła.

Pamiętajcie: dyskietki/kasety zakupione u handlarza mogą być wadliwe, programy na nich mogą nie działać itd. W takim przypadku redakcja C&A nie bierze na siebie odpowiedzialności — tu musicie już na własną rękę staczać boje z piratami.

Co innego, jeśli zakupicie/zamówicie programy u nas. Wtedy, jeśli coś będzie nie tak, macie pełne prawo do reklamacji. Wybór należy do Was...

SEKRETARZ REDAKCJI

* * *

Rozszerzenie pamięci
MEGAMIX 500

W czasach, kiedy kupowałem swoją Amigę, posiadanie IMB pamięci wystarczało do uruchomienia każdego programu oraz znacznie zwiększało komfort pracy w porównaniu ze standardową Amigą (0,5 MB). Jednak z biegiem czasu zaczęły pojawiać się programy wymagające nieco więcej pamięci. Przeżyłem prawdziwy szok, gdy podczas uruchamiania jakiejś gry zobaczyłem informację: „Jeżeli masz tylko 1 MB pamięci, wyłącz zewnętrzną stację". Do tej pory informacja podobnej treści była przeznaczona dla użytkowników standardowej Amigi. Zacząłem poważnie zastanawiać się nad sposobem „rozszerzenia się" o dodatkowe megabajty zwłaszcza, że nowe, wspaniałe programy do RayTracingu, DTP czy CAD nie chcą się w ogóle uruchomić, lub zostawiają śladowe ilości pamięci do pracy na Amidze 1 MB. Jedną z możliwości jest rozszerzenie MegaMix 500 niemieckiej firmy 3-STATE ComputerTechnik.

PRZEZNACZENIE, ZESTAW, GWARANCJA

Rozszerzenie to jest przeznaczone zarówno dla A500 jak i A500+. Pozwala na zwiększenie pojemności naszej „przyjaciółki" od 2 do 8MB RAM, „porcjami" co 2 MB. Jest ono bardziej atrakcyjne od rozszerzeń innych typów, ponieważ podłączając je do komputera zwiększamy ilość tzw. pamięci Fast, czyli tej, do której ma dostęp tylko mikroprocesor. Jakie wynikają z tego korzyści powiem później.

W skład zestawu, oprócz oczywiście rozszerzenia, wchodzi polskojęzyczna instrukcja i 12. miesięczna gwarancja. Całość jest opakowana kartonowe pudłeko.

KONSTRUKCJA

Rozszerzenie zamknięte jest w metalowym pudełku kolorem i kształtem dopasowanym do Amigi. Ma ono wyłącznik oraz diodę sygnalizującą czy jest włączone, czy wyłączone. Do najważniejszych jego zalet zaliczyć można tzw. przelotowość. Karta nie ogranicza rozbudowy systemu, pozwala na dołączenie innych modułów, np. Action Replay czy dysku twardego.

Główną część rozszerzenia stanowi płyta do której z jednej strony przylutowane jest złącze krawędziowe, a z drugiej „wystaje" szyna przelotki Expansion Port. Prostopadle do płyty głównej poprzez złącza przylutowana jest karta kontrolera i karta pamięci. Wszystkie układy mają swoje podstawki, co upraszcza wymianę w razie uszkodzenia.

ROZBUDOWA

Jeżeli nie zdecydujemy się na zakup od razu pełnej wersji (8 MB), to dalsza rozbudowa nie nastręcza problemów. Wystarczy zdjąć obudowę, przylutować kartę z dodatkową pamięcią i ustawić przełącznik konfiguracyjny zgodnie z ilością posiadanej pamięci. Jest to zresztą szczegółowo wyjaśnione w instrukcji obsługi. Kto ma wstręt do śrubokręta i lutownicy może



poprosić o instalację nowych kości przy ich zakupie.

INSTALACJA

Rozszerzenie podłącza się do szyny Expansion Port, tej z lewej strony Amigi. W sumie instalacja zajęła mi około 20 sekund, z czego większą część poświęciłem na wyjęcie zaślepki osłaniającą szynę Expansion z komputera. Znając z opowiadań sposób i czasochłonność montażu rozszerzenia z przejściówką pod układ GARY, uznałem ten czas za rewelacyjny. Poza tym w żadnym przypadku nie ma tu mowy o rozkręcaniu komputera, czy wyjmowaniu kości.

W AKCJI

Testowanie rozszerzenia rozpocząłem od programiku Avail z Workbencha, który poinformował mnie, że posiadam 3 MB pamięci (miałem 1 MB własne). Pełny sukces. Do bardziej szczegółowej inspekcji użyłem programu SysInfo. Za jego pomocą dowiedziałem się, że dodatkowe 2 MB Fast RAM zaczyna się od adresu $200000, czyli tak, jak być powinno.

Następnie sprawdziłem szybkość komputera. Według SysInfo moja Amiga była szybsza 1.33 razy od standardowej (bez rozszerzenia 1.08). Niby to niewiele, ale zawsze trochę szybciej. SysInfo podał również, że nowa pamięć jest szybsza od pamięci CHIP o 11%. To jest właśnie ta zaleta, o której wspomniałem na początku. Zainstalowanie Fast Ramu zwiększa szybkość Amigi.

Na drugi ogień wziąłem DPainta. Przy 1 MB pamięci żadną miarą nie dało się stworzyć animacji w najwyższej rozdzielczości, nawet jednej klatki. Teraz nie było z tym kłopotów. Pracowicie tworzyłem i tworzyłem, aż poddałem się przy trzydziestej klatce — po prostu zabrakło mi konceptu. W każdym razie komunikatu o braku pamięci nie zobaczyłem.

Szukając następnych programów do testowania zobaczyłem IBM-emulator. Z nim zawsze były problemy wynikające z braku pamięci. Przeważnie po uruchomieniu zostawiał mniej pamięci niż Transformer, tak że nie chciała się uruchomić nawet Eureka. Teraz rewelacja. Dla emulatora przeznaczyłem 704 KB (więcej nie można), a i tak zostało 1.8 MB wolnego dla Amigi. W takich warunkach „udawanie" wielkiego niebieskiego zaczyna nabierać sensu.

Dzięki rozszerzeniu mogłem również poszaleć sobie z KickStartem 2.0. Mimo, że zajął on 0.5 MB pamięci, to i tak zostało wystarczająco dużo na uruchomienie innych programów. Tym sposobem sprawdziłem parę programów, które żadną miarą nie chciały „chodzić" pod kontrolą Kickstart 1.3.

W okresie kiedy testowałem rozszerzenie, nie miałem zbyt dużo czasu na granie, ale te gry, które w wolnych chwilach uruchomiłem, nie buntowały się z powodów większej ilości RAM. Doszło nawet do tego, że Harpoon, moja ulubiona gra, nawet zwiększyła szybkość działania (mniej więcej dwukrotnie).

W okresie trwania całego testu nie spotkałem się z przypadkiem guru. Myślę, że należy to przypisać nie tylko prawidłowej pracy rozszerzenia, ale i korzystaniu ze sprawdzonych programów. Pogłoski rozsiewane przez użytkowników podatkowych rozszerzeń, mówiące o tym, że strasznie się one grzeją nie potwierdziły się. Owszem, nagrzewało się, ale nie tak, żeby można byłoby smażyć na nim jajecznicę. Pozwala to podejrzewać, że zastosowano układy o niskim poborze mocy, które w niewielkim stopniu obciążają „cherlawy" standardowy zasilacz Amigi.

WERDYKT

Rozszerzenie MegaMix 500 jest bardzo dobrym rozwiązaniem dla użytkowników wykorzystujących Amigę właściwie do wszystkich celów. W przeciwieństwie do innych rozwiązań, będących układami zamkniętymi, MegaMix 500 pozwala na łatwe zwiększanie ilości pamięci do pełnych 8 MB Fast RAM. Niestety im więcej megabajtów tym większa cena.

Dodatkową zaletą jest możliwość wykonania drobnej przeróbki, polegającej na przeadresowaniu pamięci SLOW (wkładanej od spodu) na Chip, co zostało opublikowane w numerze C&A 04/92. Mamy wtedy do dyspozycji 1 MB CHIP RAM i od 2 do 8 MB FAST. Jakie są tego efekty, radzę przekonać się na własne oczy.

PAWEŁ GALAS

DYSTRYBUTOR: 3-STATE POLAND
02-695 WARSZAWA, ul.Orzycka 20
tel. 431656, fax 132193
CENA (hurtowa, promocyjna): 2.390.000 zl

ZALETY:
+ Bardzo prosta instalacja
+ Łatwa rozbudowa o kolejne 2MB pamięci
+ Port przelotowy
+ Polska instrukcja obsługi
+ Gwarancja

WADY:
– niezbyt estetyczny wygląd,
– bardzo silnie świecąca dioda

TURBO • TURBO • TURBO • TURBO • TURBO



Turbo Junior II

4 mikroprzełączniki
1 przycisk fire
cena: 7.75 DM x kurs zł wg. NBP



Turbo Micro 6

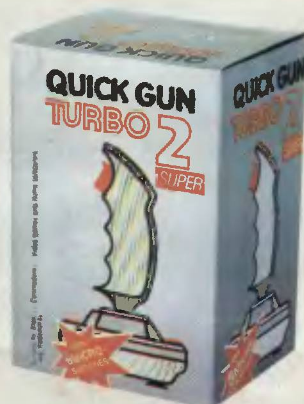
6 mikroprzełączników
1 przycisk fire
ergonomiczna konstrukcja rękojeści
cena: 10.95 DM x kurs zł wg. NBP



Turbo PRO 300

4 mikroprzełączniki
2 przyciski fire
uniwersalna rękojeść
cena: 7.25 DM x kurs zł wg. NBP





Turbo 6, Turbo 2 super

6 mikroprzełączników,
8 mikroprzełączników
ergonomiczna konstrukcja rękojeści
precyzyjny mechanizm
duża trwałość

Turbo 6 dodatkowo:
auto fire
1 przycisk fire
cena: 13.95 DM x kurs zł wg. NBP

Turbo 2 super dodatkowo:
przełącznik auto fire/normal
2 dodatkowe przyciski fire
cena: 14.95 DM x kurs zł wg. NBP



Turbo PROFI

8 mikroprzełączników
ergonomiczna konstrukcja
regulator prędkości
auto fire
kontrolka LED do auto fire
2 dodatkowe przyciski auto fire
cena: 15.95 DM x kurs zł wg. NBP



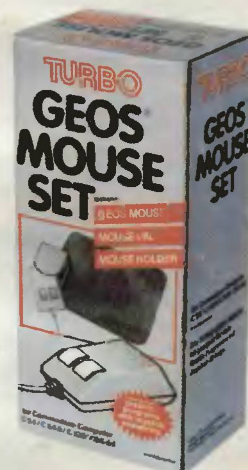
Turbo PRO, Turbo PRO acryl

6 mikroprzełączników
przełącznik auto fire / normal
kontrolka LED do auto fire
2 szybkostrzelne klawisze fire
idealny do gier sportowych
cena: 15.55 DM x kurs zł wg. NBP



Turbo IBM

analogowy joystick
do komputera IBM
precyzyjny mechanizm
wysokiej klasy potencjometry
2 przyciski fire na mikroprzełącznikach
cena: 19.95 DM x kurs zł wg. NBP



Geos Mouse Set

mysz do C-64
z uchwytem
i mouse padem
cena: 45.00 DM x kurs zł wg. NBP



Turbo Corder

magnetofon do C-64
doskonała jakość mechanizmu
solidna i wytrzymała budowa
12 miesięcy gwarancji
cena: 33.00 DM x kurs zł wg. NBP



FLOPPY 9900

najnowszy model stacji dysków do Commodore 64 w pełni kompatybilna ze stacją 1541 II
30 % szybsza
mała gabarytowo i estetyczna obudowa
cicha praca
12 miesięcy gwarancji
cena: 225.00 DM x kurs zł wg. NBP

Track Ball

To nowość zastępująca mysz
4 modele które pracują z komputerami Amiga, Atari ST i IBM
nie wymagają konserwacji
w trakcie pracy potrzebują mniej miejsca od myszki przez to są wygodniejsze w użyciu
zarówno do pracy jak i do gier
estetyczne i dokładnie wykonane
z 12 miesięczną gwarancją
cena w zależności od modelu: 69.00-89.00 DM x kurs zł wg. NBP



W sprzedaży posiadamy także: scannery do Amigi i IBM, stacje dysków do Amigi, rozszerzenia pamięci do Amigi 500 i 500 plus - 512kB 1MB i 2MB, myszy do Amigi i IBM, osłony, pudełka i wiele innych dodatków do komputerów.

Sklepy i hurtownie prowadzące ciągłą sprzedaż naszych towarów:

- „VADIM" Zielona Góra ul. Kupiecka 28 tel. 656-72
- „GRACOM" Malbork ul. Mickiewicza 26 tel. 33-14
- „Com-Studio" Łódź ul. Kilińskiego 42 tel. 333-680
- „METRO" Poznań ul. Ratajczaka 31 tel. 527-563
- „MICROMAN" Katowice ul. Karoliny 4 tel. 588-471 w. 226

Inne sklepy i hurtownie zapraszamy do współpracy.

„PROABIT"
Raszyn k/Warszawy
ul. Mickiewicza 14
kod 05-090 tel./fax 56-08-91

HURT • DETAL • WYSYŁKA

TURBO • TURBO • TURBO • TURBO • TURBO